FOR REFERENCE ONLY

INTELLIGENT MACHINE CONTROL USING NEURAL FUZZY ALGORITHMS

A thesis submitted in partial fulfilment of the requirements of The Nottingham Trent University for the degree of Doctor of Philosophy

Chi-Hsien Victor Shih

February 1996

Department of Computing, The Nottingham Trent University
Burton Street, Nottingham NG1 4BU
United Kingdom

Abstract

Conventional machine control typically requires the measurement of numerous parameters in order to define the system state. These various parameters, from sensors such as shaft encoders, accelerometers, etc., are combined through the control system to provide output actuation. For many problems, the vast number of input transducers required and the complexity of control algorithms mean that such problems are not economically addressable. This is particularly true when non-linearity of the process and time variant process parameters are encountered. In contrast a skilled operator would be able to judge, for example, the quality of cut by visual inspection of the workpiece, and based on experience adjust some of the parameters in order to improve the performance. The work reported in this thesis attempts to employ Artificial Intelligence in combination with remote sensing, in order to reduce the need for feedback sensors and achieve a more effective computer control solution.

The research uses automation of lace trimming as a suitable platform for investigation. The main problems here are cutter path detection in real-time and coping with material flexibility. The system has to work with many different patterns and sizes of lace as well as tolerating misalignment. To achieve a sufficient degree of automation, the trimming path must be located without prior knowledge of the lace pattern. A Fuzzy Reasoning Rule-based technique is applied to overcome the problem of material pattern variation and distortion. Finding the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach, namely the Line Mapping Method, is devised to speed up the search for the path. Experimental results indicate that the path can be successfully detected in different lace patterns in real time, whilst coping with lace distortion.

Work has been reported in using non-tactile means to cut deformable materials. Although the use of non-tactile cutters reduces material deformation, distortion due to mechanical feed misalignment persists. Changes in the lace pattern are also caused by the release of tension in the lace structure as it is cut. To tackle the problem of distortion due to material flexibility in general, a novel approach using inexact algorithms, i.e., fuzzy logic, neural networks and neural fuzzy technique, is developed. A Spring Mounted Pen is used to emulate material distortion caused by tactile cutting and feed misalignment. Using pre- and post-processing vision systems, it is possible to monitor the effects of flexibility and generate on-line information for error compensation. Applying the algorithms developed, the system can produce excellent results, much better than a human operator.

The system developed is a novel approach to flexible sheet material trimming and has further applications where modelling system behaviour characteristics is difficult. Such systems can range from controlling a robot moving on a slippery surface or piloting a boat. Furthermore, by relying on the intelligent software engine, problems such as transmission backlash, joint flexibility and stick-slip can potentially be compensated for. When characteristics of the mechanism, such as component wear and temperature, change over time, the controller can learn the new system behaviour and automatically make appropriate compensation. An industrially sponsored programme of work has just commenced to develop a commercial machine tool controller based on the developed principle.

The work described in this thesis is the author's own, unless otherwise stated, and it is, as far as he is aware, original.

Acknowledgement

I would like to express my gratitude to my supervisors: first supervisor, Dr. Nasser Sherkat for long technical discussions, encouragement and support in various matters of the project, and my second supervisor, Dr. Peter Thomas for making this work possible, supporting the research project and providing different views of the problem and inspiring discussions.

I would also like to thank Mr. David Moore and Mr. Tim Sellears who gave me complete technical support on software and hardware.

Finally, thanks to the Nottingham Trent University, Pacer Systems Ltd. and Axiomatic Technology Ltd., without whose support this project could not have been carried out.

To my parents,
With love and thanks for their support and encouragement

Table of Contents

List of Illustrations

CHAPTER 4. LACE PATTERN DETECTION

CHAPTER 5. TIGHTLY COUPLED VISION AND CONTROL SYSTEM

CHAPTER 7. DISCUSSION

APPENDIX A. TELECENTRIC LENS

List of Tables

Glossary of Terms

Activation function

Algorithm for computing the activation value of a network node as a function of its net input. The net input is usually the sum of the weighted inputs to the neurode, but may take on many other forms.

AI

Artificial Intelligence.

ANN

Artificial Neural Networks.

Back-propagation

A learning rule for multilayer feedforward networks, in which weights are adjusted by backward propagation of the error signal from outputs to inputs.

Batch training

Procedure for training neural networks, in which the weights are adjusted for each epoch.

Bias

Weight from an unit in a neural network that is always on. Acts on a network unit like an offset. All units in a network, except those in the input layer, usually have a bias.

Binary image

A black and white digitised image represented as zeros and ones.

Bitmap

A binary image represented by zeros and ones.

CAD

Computer Aided Design.

CAM

Computer Aided Manufacturing.

CCD Camera

Charge coupled device camera.

CCIR

European monochrome broadcasting system standard which requires 625 lines at a 50-Hz frame rate.

Defuzzification

The process of converting fuzzy outputs into a single raw or crisp output.

Digital image

A representation of a visual image by an array of brightness values.

Edge

A change in pixel values (exceeding some threshold) between two regions of relatively uniform values. Edges correspond to changes in brightness which can correspond to a discontinuity in surface orientation, surface reflectance or illumination.

Edge detection

The ability to determine the true edge of an object.

Epoch

Presentation of a set of training patterns to an Artificial Neural Network.

FAM Bank

Fuzzy Associative Memory bank, which is used to reduce the number of the fuzzy rules for speeding up the calculation.

Frame

Digital data representing a single image at a specific point in time.

Fuzzification

The procedure of calculating an input value to represent a degree of membership in one or more fuzzy sets.

Fuzzy Engine

An intelligent software kernel based on applying fuzzy logic technique.

Fuzzy Logic

Fuzzy logic is an extension of set theoretic multivalued logic in which the truth values are linguistic variables (or terms of the linguistic variable truth).

Fuzzy Set

A fuzzy set is a set containing elements which have varying degrees of membership in the set. Elements in a fuzzy set can be members of other fuzzy sets on the same universe.

Gradient decent

Algorithm for minimising some error measure by making small incremental weight adjustments proportional to the gradient of the error.

Gray level (scale)

A quantised measurement of image irradiance (brightness), or other pixel property usually given in integer values.

Gray scale image

An image consisting of an array of pixels where each pixel has a value representing the average light intensity on the area. Typically, 16, 64, or 256 levels are possible for each pixel, depending on the number of bits available to process and store data.

Histogram

Frequency counts of the occurrence of each intensity (gray level) in an image usually plotted as the number of pixels with a given gray value vs. gray level.

Image enhancement

The use of processing techniques to accentuate certain properties to improve the nature of the information received from an image.

Image processing

Transformation of an initial image into a second image with more desirable properties, such as increased sharpness, less noise and reduced geometric distortion.

Inexact Algorithms

An innovative approach which applies fuzzy logic, neural network and neural fuzzy technique as an intelligent kernel for solving very complex problems.

Intensity

The relative brightness of an image or portion of an image.

Learning rate

Parameter that regulates the relative magnitude of weight changes during learning.

Learning rule

An algorithm for adjusting the weights and connections of a network based on experience.

Least mean-square rule

Window-Hoff rule. Learning rule in which change of weight is proportional to the difference between the actual activation and the desired activation. The rule leads to minimisation of mean-squared error.

Machine vision

The ability of an automated system to perform certain tasks normally associated with human vision, including sensing, image formation, image analysis, and image interpretation or decision making.

Membership Function

If X is a collection of objects denoted generically by x then a fuzzy set $\tilde{A}$ in X is a set of ordered pairs: $\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}$. $\mu_{\tilde{A}}(x)$ is called the membership function, or grade of membership of x in $\tilde{A}$ which maps X to the membership space M (0, 1).

Momentum factor

Constant used to promote stability of weight adaptation in a learning rule. To prevent excessive weight change and possible oscillation, weight changes are moderated by a term that is proportional to the previous weight change and the momentum factor.

Network paradigm

Particular choice of a set of network attributes to achieve a particular kind of processing.

Neural Engine

An intelligent software kernel based on applying neural network approach.

Neural Fuzzy Engine

An intelligent software kernel based on applying neural fuzzy theory.

Neural Networks

Networks of adaptable nodes which, through a process of learning from task examples, store experiential knowledge and make it available for use.

Neurode

Active unit in a neural network. Consists of a set of inputs from other neurodes and an output that can go to any number of other neurodes. Performs an activation function on its inputs to produce an activation value that is placed on the output. Can contain local storage that is used to compute the activation value. A neurode is activated by the update procedure.

NTSC

Standard for colour image in America which augments the RS-170 standard by utilising a 3.58 MH colour sub carrier on the video signal.

PAL

Standard for colour images in the European broadcast system which is the equivalent of NTSC in the American system.

Path-following-error

The difference between the desired path and the actual path.

Perceptron

Simple network developed by Roseblatt, consisting of an input layer connected to a single neurode. The activation function of this unit is a linear threshold function applied to the inner product of the input and weight vectors.

Pixel

The smallest element of a scene, a picture element, over which an average brightness value is determined and used to represent that portion of the scene. Pixels are arranged in a rectangular array to form a complete image of the scene.

Probability

The term of *probability* in the document denotes a confidence level.

Resolution

The smallest feature of an image which can be sensed by a vision system. Resolution is generally a function of the number of pixels in the image, with a greater number of pixels giving better resolution.

RS 170

Standard TV monitor interface; 525 scans, interlaced, 1/30 second, active time 52 microseconds, retrace time 11.6 microseconds.

Sigmoid function

Nonlinear activation function whose output is a non decreasing and differentiable function of the input with maximum and minimum saturation values.

Spring Mounted Pen (SMP)

A pen with a spring on the top connected with the Z axis of a cutting mechanism. This mechanism is used to emulate the movement (distortion) of the material strip due to the cutting forces caused by a tactile cutter.

Squashing function

Function whose value is always between finite limits, even when the input is unbounded.

Sum-squared error

Measure of total error of a network for a given set of input target pairs.

Supervised learning

Learning procedure in which a network is presented with a set of input pattern target pairs. The network compares its output to the target and adapts itself according to the learning rules.

Thresholding

Separating elements or regions of an image for processing based on pixel values above or below a chosen (threshold) value or gray level.

Unsupervised learning

Learning procedure in which a network is presented with a set of input patterns. The network adapts itself according to the statistical associations in the input patterns.

Weight

Connection strength. Strength of a connection between two neurodes, which determines the amount of effect that one neurode can have on the other. Connections have a positive, zero, or negative weight. Positive values are excitatory, and negative values are inhibitory.

1. INTRODUCTION

Chapter 1
INTRODUCTION

1.1 Machine Vision System

In manufacturing industry, tasks requiring the use of eyesight have in the past always been carried out by human operators. This restricted the spread of automation mainly to repetitive mechanical tasks, working with materials of good and consistent quality, or to those where variations are closely controlled and the necessary feedback of information from the process is of a fairly simple kind.

Machine vision is both better and worse than eyesight. In certain circumstances it can be faster. It will continue working round the clock and will not tire. It can work in conditions which would be very unpleasant or impossible for a human operator. It can take dimensional measurements better than a person can estimate by eye, and can give an objective measure of other variables such as colour which an inspector could only assess subjectively. On the other hand machine vision is not backed by the power of the human brain, it needs time to analyse a complex scene or to look for several characteristics. It is fussy about lighting and may have difficulty in coping with reflections and with insignificant random variations in the objects it is looking at. At present there are two broad areas of application for vision systems in industry:

1) Inspection - This is at present by far the most important area [HOL84] and there is a great diversity of application, including not only the visual examination for defects such as normally done by eye but also measurement of dimensions or other characteristics, counting, checking of orientation, and so on.

2) Guidance - Applications include control of robots, vehicles, orientation devices and so on in parts handling, sorting and transport.

The benefits of using machine vision are as follows:

1) No alternative - There are some situations where vision has provided a solution to a problem where no alternative has been found feasible, such that it has been the means of making a new production process possible;

2) Safety and reliability - Where human life is at stake, whether it is in handling dangerous drugs or potentially lethal vehicles, every item must be inspected at least once, sometimes more often, and even then there may be sample testing of finished products;

3) Product quality - Below the absolute demands of safety there is the large and increasingly important area of product quality. In a highly competitive world, quality is often as important a consideration as price;

4) Automation aid - If machine vision is instrumental in the introduction of flexible automation, then it can claim an important share in many tangible and intangible benefits, like reduced work in progress, shorter door-to-door time, possibly a change from manufacturing for stock to manufacturing against orders, shorter lead time from design to production, and capital savings through the purchase of adaptable plant and tooling with a longer life than dedicated equipment.

Widespread use of robotics and CAD/CAM in the industrial sector provides a need for an automated process of acquiring vision information in digital form. At the same time,

social, industrial, and economic changes mandate a need to increase productivity in the manufacturing sector of the economy, improve levels of quality and reliability in the finished product, and provide better traceability of parts going into the item being manufactured for product-defect liability aspects.

1.2 Scope of the Project

Conventional machine tool control techniques use sensors such as shaft encoders and piezoelectric transducers to provide the necessary feedback signals. Such sensors supply local information on angular displacement of a rotating shaft or deflection of a machine membrane. This information is combined and used to control and monitor the operation of machine tools and robots. Although conventional techniques provide a reasonable means of control and monitoring, their effectiveness is very low. This is mainly due to the complexity of the signal combination method (control loop) and the high cost of feedback sensors.

When conventional sensors alone are used, a fairly complex technique would be required to determine whether there is a fault such as a blunt cutter head. In contrast a skilled operator would be able to judge the quality of cut by visual inspection of the workpiece. By employing remote sensing (camera) to monitor the performance of the machine, at the end-effector point, as well as feeding back the conventional signals, it should be possible to achieve a more effective solution [PRO90].

In such a situation, there will be no need to resort to complex methods to derive the information from the local sensors. In applications such as flexible material processing, e.g. lace cutting where visual effects of the finished product are more important than their engineering precision, it may be possible to reduce the use of conventional feedback sensors and rely almost entirely on remote sensing. Furthermore, it would be interesting to determine the effect of eliminating local sensors altogether and to rely on remote sensing and post-process monitoring alone.

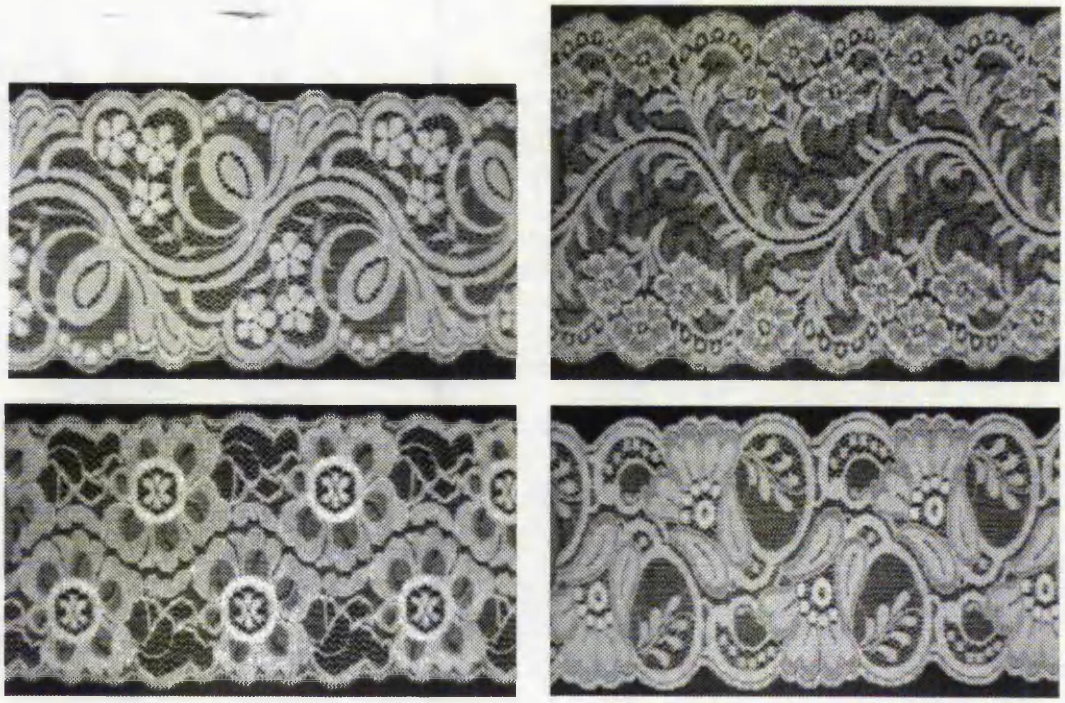


Figure 1-1: *Samples of lace patterns*

This project uses automated lace trimming as a good example for experimentation and assessment of the effectiveness of using remote sensing for control and monitoring. Lace is difficult to cut automatically. It comprises a fine and intricate pattern, with various densities of knit and holes (see Figure 1-1). On most designs the pattern repeats many times across the width and length, but in practice the repeats are never absolutely identical. Furthermore, lace is flexible, extensible and easily distorts, effectively changing the pattern [NOR91]. Norton-Wayne experiences this problem and states that "this characteristic of lace makes it impossible to cut a consistent position" [NOR91]. Russell et al. approach this problem by trying to locate a reference feature in the lace motif so that they can keep track of the changes in the pattern due to stretch [RUS88]. Moreover, the vision system has to work with many different lace patterns and sizes and tolerate misalignment, stretch and other distortions of the lace [SHE94a]. By coupling the pre and post processes (Figure 1-2), it is possible to monitor the process and generate on-line information for the controller and hence overcome the flexibility problem.

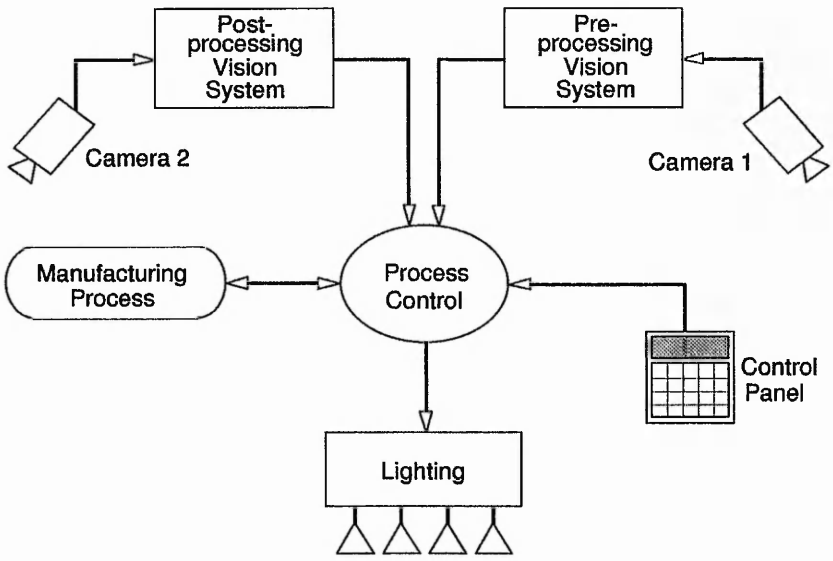


Figure 1-2: *Integrating the pre and post processing visions with a industrial production system*

In this programme of work it is proposed to use computer vision to find the cutter path and monitor the effectiveness of path finding and cutting processes downstream of the cutter. A closely coupling between the two allows detection of errors and alteration of control parameters on line.

The main emphases of the project is to investigate the use of inexact algorithms, e.g., *fuzzy logic*, *neural networks*, and *neural fuzzy technique*, in order to overcome the problems of complexity and flexibility. Furthermore, it is proposed to use the inexact algorithms as the basis for achieving optimal quality which satisfies visual demands rather than engineering precision. Employing inexact matching techniques, potentially allows development of methods for detecting unsatisfactory or faulty operation which may be applicable to similar manufacturing processes such as sign cutting. Multi-processing techniques may be use to satisfy the real-time requirements for the coupling.

1.3 Literature Review

The first published paper for the automatic cutting of deformable materials using machine vision was in 1988 [RUS88]. This lace cutting system employs a low-cost binary vision system which uses a Micron Technology IS32 DRAM as its optical sensor to give a 256 by 64 pixel binary image. An Intel 8751 single-chip microcomputer interfaces directly with the IS32 vision sensor. This microcomputer performs the image processing operations and transfers the results to a BBC personal computer for display. The BBC personal computer also issues commands to select the operations performed by the vision system. A back lighting structure is selected in the system.

A form of template matching is used to determine the position of the lace in the vision system image. A 'slice' across the width of the lace pattern, at a point which includes some distinguishing feature, is recorded as a template. By examining the difference between this template and the lace image, the extracted information can be used to determine the actual position of the lace image. According to the experimental results mentioned in the paper [RUS88], the 8051 microcomputer takes about *2 seconds* to perform a template match over the full area of the image. The exact time depends upon the precise location of the lace within the field of vision.

The experimental results indicate that the speed for detecting a cutting path is insufficient for a commercial type of machine. As template matching is used to determine the actual position of the lace image, prior knowledge for each lace pattern is required before the system is operated.

In 1992, another computer vision based lace scalloping system was reported [BRI92] [KIN93]. This system was designed for tracking *pre-defined path* along a patterned web of material. The system uses multiple digital signal processors (DSPs) to acquire and process image data from a high-resolution line-scan CCD camera. Control information is generated

to allow cutting of the web along the tracked path using a CO_2 laser beam defected by a galvanometer mounted mirror. Illumination of the system is achieved by back-lighting of the web along the line of view of the camera.

The vision system comprises a high speed, high resolution linescan camera (0.1 mm and 0.35 mm in each direction) coupled via a specifically designed interface to two LSI VME DSP boards with multiple Motorola 56001 devices (Figure 1-3). The DSP boards communicate with the supervisory computer (68020 processor) over the VMEbus. The linescan camera is a 2K element CCD device with specifications sufficient to allow a spatial resolution of 0.1 mm. The camera's field of view is 200 mm wide. The interface board allows the first DSP to set the threshold of the image data and also compiles 24 bit data packets which are then transferred into shared memory. The lace movement is sensed by a high resolution optical encoder. The second DSP in the system is responsible for the tracking of the material and derivation of the control signals for the laser cutter. The basic technique is to use a pre-defined *reference map*. The reference map is made by scanning one

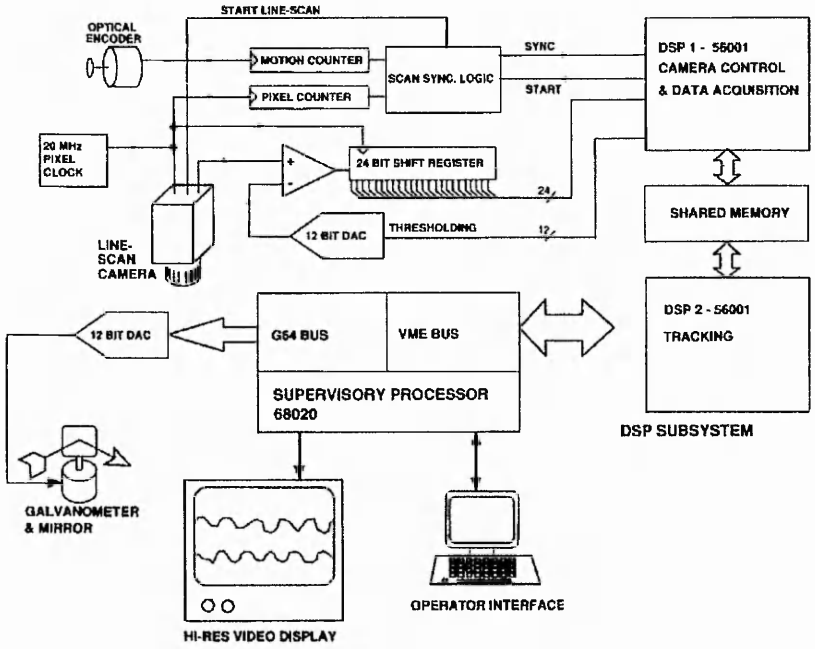


Figure 1-3: *The hardware structure of the King's lace scalloping system* [KIN93]

pattern repeat and manually defining the required cutting path using the high resolution display. This is required for each new material sample and the results are stored on disk.

Centre-weighted line match (incremental algorithm) and decaying impulse response of a particular filter [KIN93] are used to perform the matching and tracking processes of the lace pattern. This approach is robust against distortion and scale errors of ±10% in both directions across the web. The results of the above processing are placed in a FIFO buffer ready for passing to the laser positioning controller. According to the experimental results stated in the report, scalloping is successfully carried out at web speeds of 220 mm/s (13.2 meter/minute). This is limited primarily by the processing capability of the DSP performing the tracking algorithm.

Similar to the previous case, template data from one repeat of the lace pattern is defined manually as a reference map for tracking the actual cutting path. This is considered a disadvantage from the automation point of view.

1.4 Outline of the Thesis

This report is structured into eight chapters.

Chapter 1 introduces the machine vision system and the main goals of the project. The present techniques developed for deformable material processes are reviewed.

Chapter 2 presents the vision equipment employed in the system and the techniques used to process the captured images. Methods to correct the optical distortion are also discussed.

Chapter 3 briefly describes the selected test rig, presents the methods to calibrate the camera with this CNC machine and the scheme to manage the operation of the vision based control in real-time.

Chapter 4 presents the real-time automatic lace trimming system. The fundamentals of fuzzy logic theory are described. A number of methods are designed to detect the cutting path from a lace pattern, including the pixel intensity directed feature extraction approach, fuzzy pattern recognition technique and line mapping method. The scheme to supervise the vision system and the cutting mechanism is presented. Various experimental results are provided to verify the developed methods.

Chapter 5 defines a coupled vision based control system using inexact algorithms, such as fuzzy logic, neural networks and neural fuzzy technique. Through a learning process, the intelligent system can automatically detect and correct the processing errors, e.g., path-following-errors. The theories of neural networks and neural fuzzy system are introduced here.

Chapter 6 describes a new type of learning technique - the rapid learning algorithm. Only one frame of the training process is required to minimise the deviation between the intended pattern and the actual drawing pattern. The scheme of engaging a neural-network-based kernel to create the compensated pattern to the controller is presented.

Chapter 7 discusses the work conducted and some problems encountered during the experiments. Future work is suggested.

Chapter 8 concludes the thesis.

Five appendices are provided at the end of this document:

Appendix A provides information on the telecentric lens for the vision system.

Appendix B lists the PACER VMC HPGL commands used in the project.

Appendix C illustrates various samples of SMP following processes using the 2VMethod (together with the Piecewise Error Compensation Algorithm) and the 3VMethod, respectively.

Appendix D illustrates a number of samples of SMP following processes using the Generic Error Compensation Algorithm.

Appendix E provides copies of papers published by the Author.

2. VISION SYSTEM AND IMAGE ANALYSIS

Chapter 2
VISION SYSTEM AND IMAGE ANALYSIS

2.1 Introduction

Machine vision and digital imaging technology is multi-disciplinary in the sense that the field uses the knowledge of traditional engineering and computer programming for the different parts of the process. The process can be subdivided into the following three activities:

1) Obtaining the digital representation of an image;
2) Employing computational techniques to process or modify the image data;
3) Analysing and using the results of the processing for the purpose of guiding robots or controlling automated equipment, assuring a level of quality in a manufacturing process or supporting statistical analysis in a computer-assisted-manufacturing (CAM) system.

A machine vision system comprises all the elements necessary to obtain a digital representation of a visual image, to modify the data and to present the digital image data to the external world. The system may appear complex in an industrial environment due to all

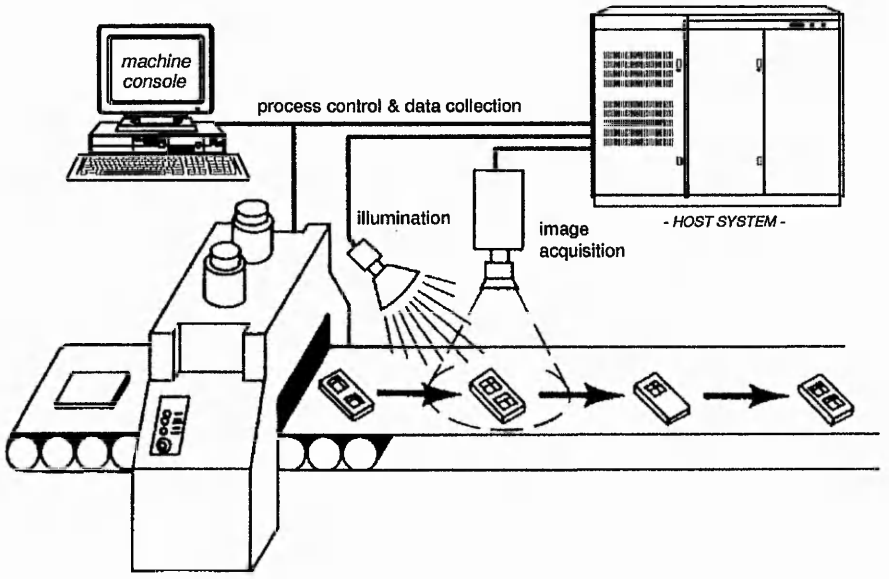


Figure 2-1: *Industrial vision based manufacturing system*

the associated manufacturing process equipment used in the application. The complexity is reduced when the system is viewed in terms of the three main functional components:

1) Image acquisition;
2) Processing;
3) Output or display.

Factory automation applications involve the use of vision technology in the inspection tasks to improve the quality of the products produced, in the data collection tasks for inventory and management control, and in the process or machine control tasks for improving manufacturing productivity. A simple industrial vision system used for factory automation could be characterised by a single camera monitoring an assembly line as shown in Figure 2-1. The vision system observes the object, determines if it is within specifications and generates command signals according to the determined results. The image acquisition equipment includes the lights, camera and the frame grabber. The processing equipment includes both hardware and software in the vision processing unit and the output equipment is the electronics interfacing the system to various parts of the manufacturing world.

A machine vision system's image acquisition time depends on the size of the image matrix, the processing time of the frame-grabbing electronics and the type of the camera. Tube type cameras operating in a conventional RS-170 mode will produce 30 images per second for standard commercially available monitors; the number of images per second could be increased by an estimated factor of five or ten by using a non-RS-170 mode. Solid state cameras can acquire the image as ten microseconds; the time required to read out the signal from the sensor will depend on the size of the matrix, processing speed and system bandwidth.

Image acquisition using machine vision systems is as much as ten times greater than of the human vision system. This ratio is increasing with time as the state of the art in electronics improves, while that of the human system is not changing. The task response capability of the machine vision system is on the order of fifteen times greater than of the human system [LOU90]. In this respect, a machine vision system is potentially allowed to achieve a better performance than a human operator.

2.2 Configurations of the Vision System

The proposed machine vision system mainly consists of five elements: 1) Frame Grabber, 2) Camera and Lens, 3) Video Multiplexer, 4) Analog Monochrome Video Monitor, and 5) Illumination, which are closely integrated with a host system (Intel 80486 processor running at 66 MHz).

• Frame Grabber

A VFG-512 Frame Grabber is used in the project. The basic function of the VFG-512 Frame Grabber is a plug-in card for the IBM PC. It has 256K bytes of frame memory organised as 512x512x8 bits. The VFG Frame Grabber digitises the incoming video signal

to 8 bits resolution at a rate of 30 frames per second from a standard RS-170 video input in real time. With the 8-bit A/D converter, the pixel resolution increases four fold to 256 gray levels. Each pixel uses 8 bits of storage in the frame memory, where the pixel may take a value between 0 and 255 with value 0 corresponding to the black and 255 to the white level. Image data stored in the on-board frame memory can be enhanced or modified by image processing techniques for displaying on the video monitor. Besides, data in the frame memory can be stored onto disk for later use. The image acquired can be displayed simultaneously on an analog video monitor.

The VFG Frame Grabber hardware consists of four major sections as shown in Figure 2-2. The Bus Interface section of the host computer decodes the frame memory access and I/O commands from the PC bus. The Frame Memory Subsystem consists of a dual-ported frame memory and a frame memory controller. The Digitisation Logic section digitises the incoming RS-170 interlaced video signals to 8-bit resolution and stores them into the frame memory. A phase locked loop circuit is used to synchronise the digitisation clock with incoming video signals. Also, a 10 MHz pixel clock is generated for digitising and displaying the image. The Display Logic section generates RS-170 or CCIR interlaced video output signals by reading frame memory contents at a rate of 10 million pixels per second. Any of the four 256x8 bits output Look Up Tables can be selected for pixel transformation prior to

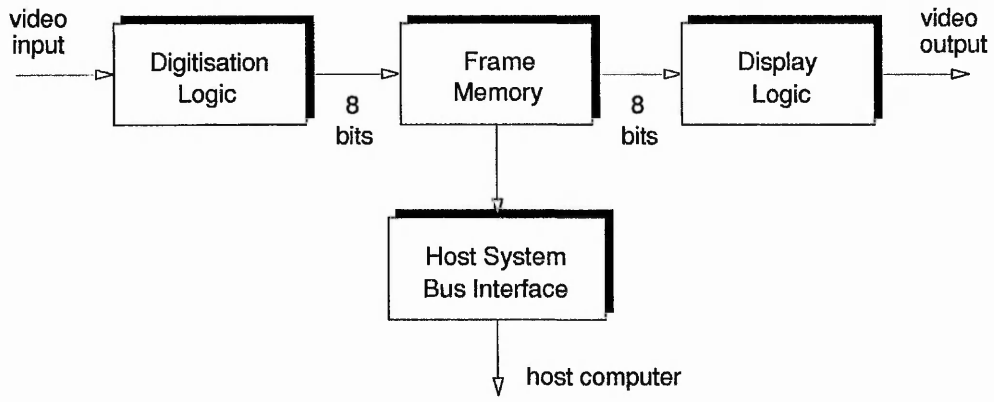


Figure 2-2: *The VFG Frame Grabber block diagram*

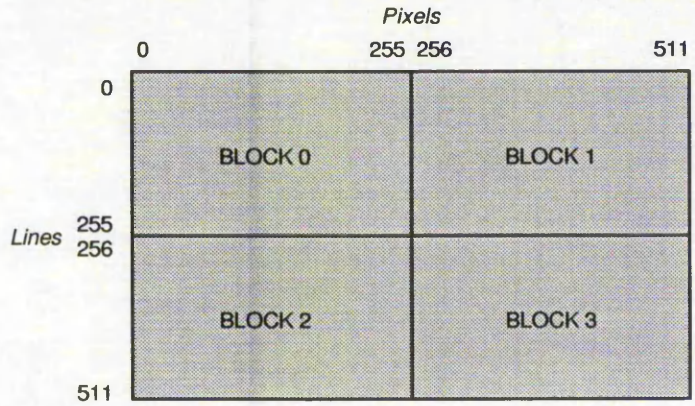


Figure 2-3: *The VFG memory block partition*

displaying on a video monitor.

As depicted in Figure 2-3 the VFG frame memory is organised as 512 by 512 pixels which are divided into four 64K byte memory blocks and mapped to four quadrants on the monitor screen, so only one block of the frame memory can be accessed by the PC at a time. Any of the sixteen 64K bytes memory address slots in the PC can be jumper selected as the mapping slot for the VFG frame memory. The frame memory is dual-ported, so the PC can access the frame memory randomly and it has a scan mode which allows simultaneous image acquisition and display. However, the image displayed has a time lag of one frame period.

· Camera and Lens

Two TM 526 Black and White CCD (Charge Coupled Device) Cameras (PULNIX) are connected to the frame grabber via a Video Multiplexer. Two different sizes of standard 2/3 inch lenses (RS) are mounted on the cameras:

- 8.5 mm F1.3 lens (pre-processing vision system);
- 16 mm F1.4 lens (post-processing vision system).

• Video Multiplexer

A VIDMUX-4 (solid state, quad video multiplexer, ImageNation Corp.) board is used in the vision system to switch four input video signals to the frame grabber. Dip switch selectable port address allowing up to sixteen vidmux - 4 boards in one computer. A simple command line written in C programming language can operate the video multiplexer to switch between different channels.

• Analog Monochrome Video Monitor

A Hitachi VM-920 Monochrome Video Monitor is engaged to receive the video signal from the frame grabber and continuously display the black and white gray-level pictures. This Monochrome Monitor is used to display the real-time image when the target object is arranged within the field of vision.

• Illumination

Illumination is a key parameter affecting the input to a machine vision system since it directly affects the quality of the input data and may require as much as 30% of the

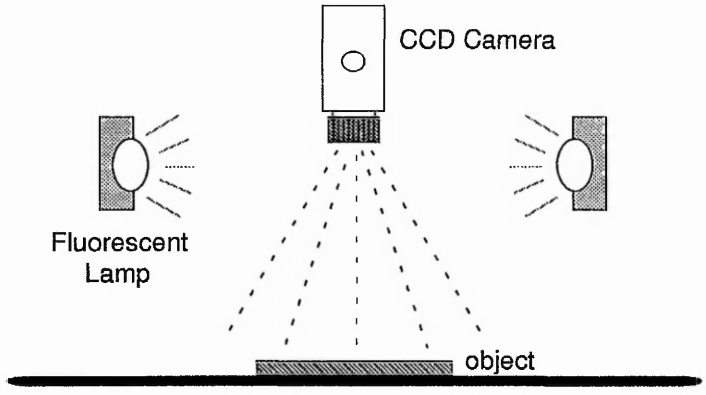


Figure 2-4: *Schematic diagram of the front lighting vision system*

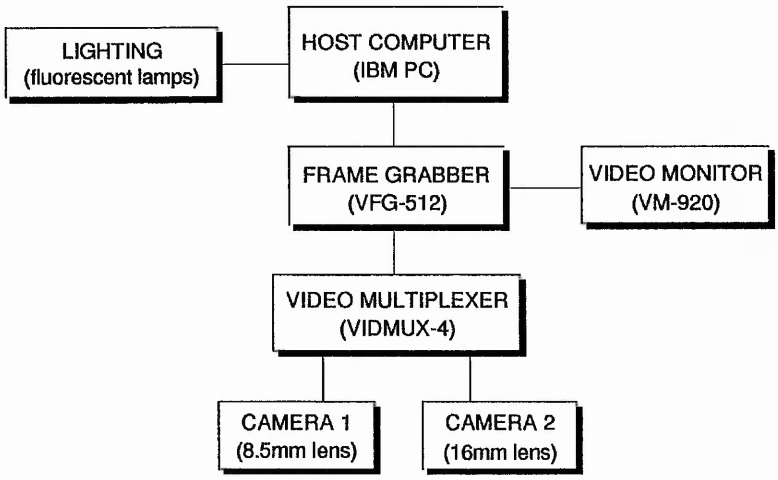


Figure 2-5: *Block diagram for the vision system overview*

application effort. It is necessary to customise the illumination for each application since there is essentially no standardised general purpose machine vision illumination equipment. The method and specific source of light energy affect the amount of processing and achievable results.

The methods for industrial applications can be subdivided into four categories: 1) back lighting, 2) front lighting, 3) structured lighting, and 4) strobe lighting. As illustrated by Figure 2-4, front lighting which employs light reflected from the object is applied to construct the illumination system in the project. The illumination sources and the camera are both on the same side of the object. This method of illumination is used to obtain information on the surface features as well as for dimensioning. A block diagram which is shown the interrelation among these vision elements is illustrated in Figure 2-5.

2.3 Thresholding Operation

Thresholding operation or bi-leveling operation, is used to remove the gray-level trends in an image, to make the gray-level regions more discrete, to 'segment' (or split into distinct parts) an image. Thresholding normally refers to setting all the gray levels below a certain

level to zero; or above a certain level to a maximum brightness level. The maximum brightness will be 255 on an 8-bit plane system. This will screen out unwanted variations in an image where all those variations are around, above or below a certain grey level.

Thresholding can be used to create a binary image (bitmap). Rarely is it possible to identify a perfect gray-level break. Normally errors are made in the classification of pixels as background or foreground as soon as any thresholding is done. When classifying a range of gray scales into one set, two types of error can be made:

1) Not all pixels are caught that should be included;
2) Some pixels caught should not be in the group.

The choice of threshold level aims to balance these two types of error, but there are circumstances when it is more favourable for there to be more of one than the other. The threshold function operator may be applied on a global scale to the entire image, or different threshold values may be used for different objects and regions of the image. Two different methods have been attempted in the project to bi-level a 256 gray-level image. The first approach is based on histogram shape analysis. The second method examines the average intensity of each vertical strip of an image.

2.3.1 Histogram analysis

Many scenes, in industrial applications for instance, consist simply of an object on a uniform background, or perhaps a slowly varying background. Such a scene is very straightforward to segment simply by looking at its histogram. A histogram is a graphical representation of the frequency count of the occurrence of each intensity (gray level) in an image. The abscissa or x-axis is the values of gray levels and the ordinate or y-axis is the number of pixels having that gray level. The histogram is constructed by

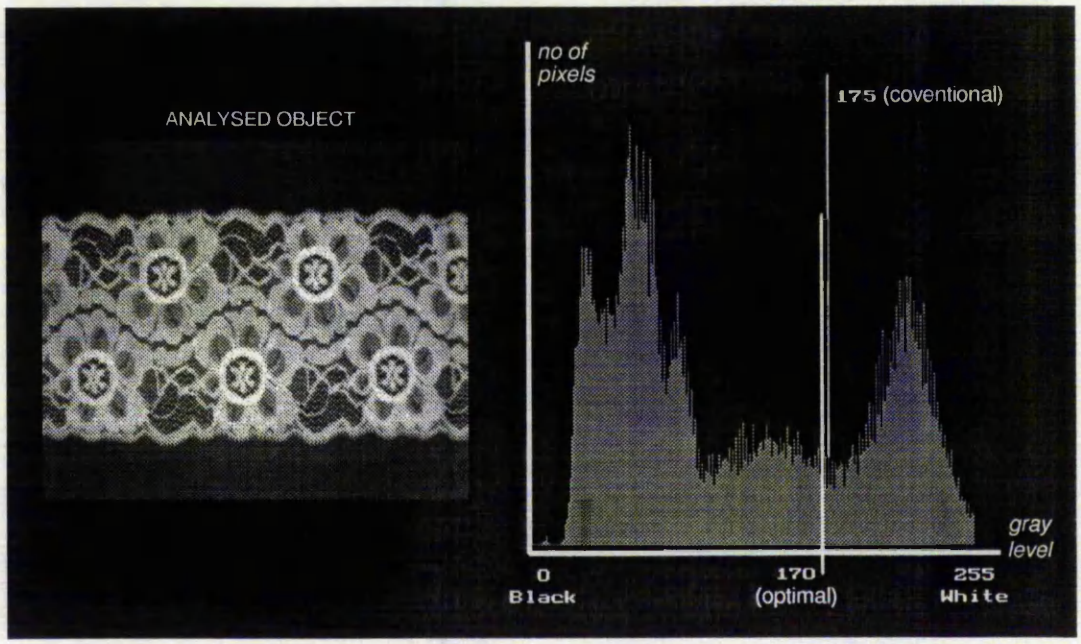


Figure 2-6: *Gray level histogram analysis using optimal thresholding*

1) Digitising the image frame;
2) Counting the pixels at each gray scale level;
3) Plotting the frequency count of pixels at each gray level.

If an image consists of an object of approximately the same gray level that differs from the gray level of the background, the resulting histogram is bi-modal. Pixels of objects form one of its peaks, while pixels of the background form the second peak. A method based on

Figure 2-7: *Bi-level lace bitmap image (using optimal thresholding)*

approximation of the histogram of an image, using a weighted sum of two or more probability densities with normal distribution, represent a different approach called *optimal thresholding*. The threshold is set as the closest gray level corresponding to the minimum probability between the maximums of two or more normal distributions, which results in minimum error segmentation [SON93].

Figure 2-6 shows the histogram analysis using the optimal thresholding. The difficulty with this method is in estimating normal distribution parameters together with the uncertainty that the distribution may be considered normal. The difficulties may be overcome if an optimal threshold is sought that maximises gray level variance between objects and background.

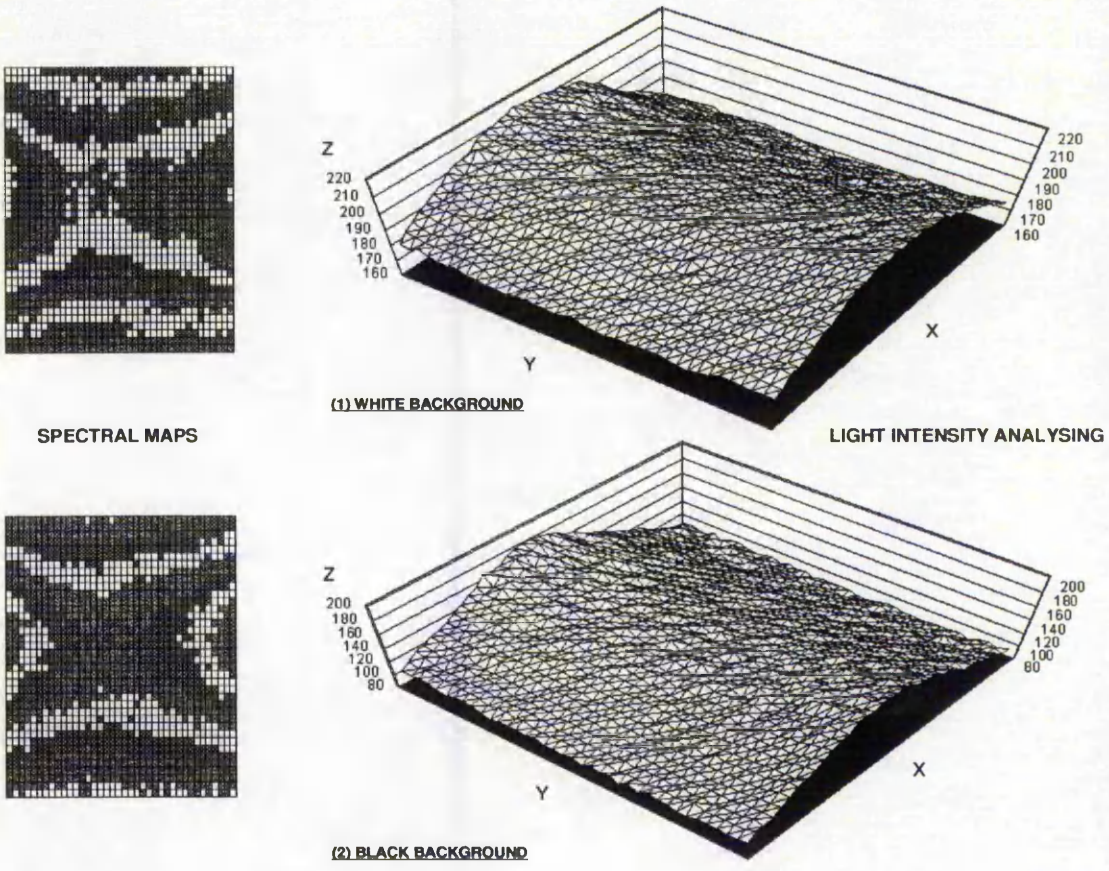


Figure 2-8: *Spectral map analysis using white and black backgrounds*

As illustrated in Figure 2-7, after the bi-leveling operation, using optimal thresholding the image shows up as a bitmap (2-colour black and white image). Inspecting the resulting bitmap, it is clear that this thresholding process is unsuitable for this problem. By analysing the *spectral map* of the background image (no object within the scene), the irregular light intensity of the illumination system is highlighted. Since two fluorescent lamps are employed to build the front lighting system (for cost effectiveness), a continuous change of the lighting all over the entire scene has occurred (see Figure 2-8, where the X and Y show the pixel grid; the Z axis shows the pixel gray-level). Using the optimal thresholding operation, which adopts only one thresholding level, normally produces dissatisfied outcomes for the project.

A new bi-leveling method based on examining the average intensity of each vertical strip of an image has been developed to solve the above problem. This method is described in the following section..

2.3.2 Average intensity analysis

This method is based on finding the average intensity of each vertical strip and examining which points lie above and below this threshold. Equation 2-1 represents the calculation of the thresholding points over the entire image.

Figure 2-9: *Bi-level lace bitmap image (using average intensity analysis)*

$$Thresholding\ Point(i) = \frac{\sum_{y=0}^{j}(x,y)\Big|_{x=i}}{(j+1)} \qquad (2\text{-}1)$$

where i = 0 to 255, and j = 0 to 255 on an 8-bit plane system.

Figure 2-9 illustrates the bi-leveled image using average intensity analysis. Compared with the binary image presented in Figure 2-7, it is clear that this new thresholding method can produce better results than using the conventional histogram analysis (optimal thresholding).

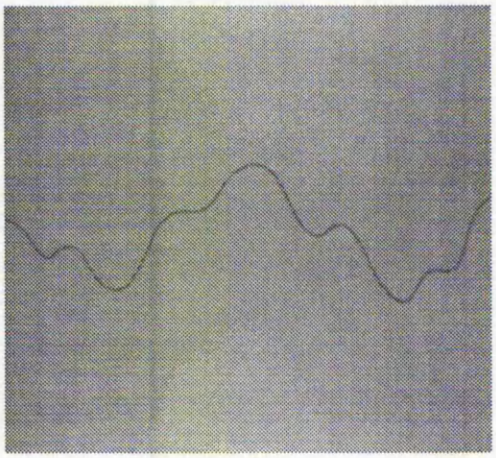


Figure 2-10: *A line captured by the vision system*

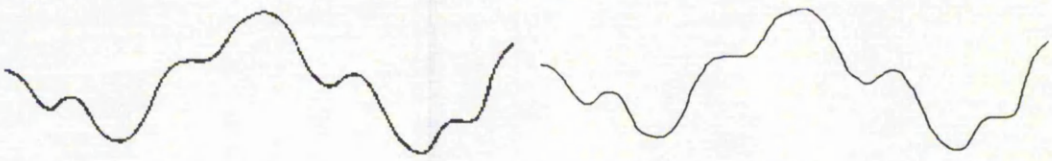
(a black line on white paper)

2.4 Image Extraction and Analysis

In the project three types of objects are analysed for producing the movement data to a cutting mechanism: 1) black line on white paper, 2) pattern on white paper, and 3) lace web. Various image processing techniques are applied in order to distinguish (extract) the object from the raw image. In the following sections, the author describes the methods for extracting a line (Section 2.4.1) and a pattern (Section 2.4.2) on white paper. The technique used to obtain the cutting path from a lace web is presented in Chapter 4.

2.4.1 Line extraction

As an image of paper is captured by the vision system (illustrated in Figure 2-10), the data is transferd to the host system and stored in the memory. The *average intensity analysis* is applied to transform the image into a binary bitmap. The small amounts of 'noise' in the binary image can be removed by looking for two consecutive points from neighbouring pixels (both 3x3 and 5x5 square windows are used) [SON93]. The *line skeleton operation* is then applied to create a skeleton of the extracted bitmap pattern. The line skeleton operation



(a) bi-leveling and noise reduction (b) line skeleton operation

Figure 2-11: *Line extracting operation*

consists of thinning the region of a pattern without losing the essential shape. Figure 2-11 illustrates the two main stages of extracting a *path* from a gray-scaled image (Figure 2-10).

2.4.2 Pattern extraction

As a pattern image is captured by the CCD camera (see Figure 2-12), it is transformed into a black and white bitmap. After removing noise within the image, a *border following process* is engaged to find the cutting path around the pattern. The border following operation is a method to outline a pattern image by following the edge around the object and back to the beginning (if the object in an image has a continuous edge all around it). As depicted in Figure 2-13, the border of the pattern has been successfully detected.

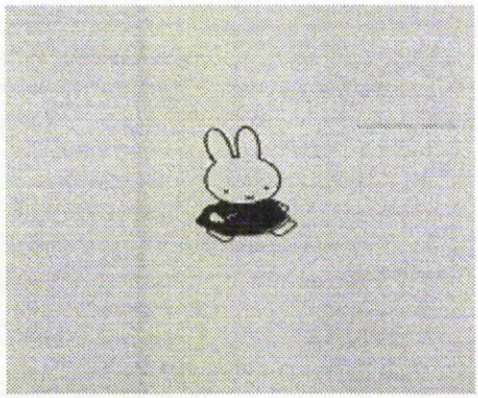
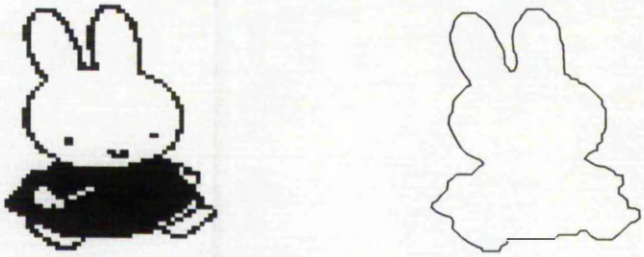


Figure 2-12: *A pattern image taken by the camera*

(a rabbit on paper)



a) bi-leveling and noise reduction b) border following process

Figure 2-13: *Pattern extracting operation*

2.5 Correction of Optical Distortion

Most conventional cameras which use fish-eye lenses create view angle and magnification errors while providing high resolution and contrast. The methods to correct the distorted image can mainly be divided into two areas: 1) use of software filter, 2) using a special camera lens.

Figure 2-14 illustrates a grid pattern captured by a CCD camera using 8.5 mm F1.3 lens. This kind of wide angle lens is a relatively short focal length lens which has a large angular field of view. When used to view an object in a machine vision application, wide angle lenses

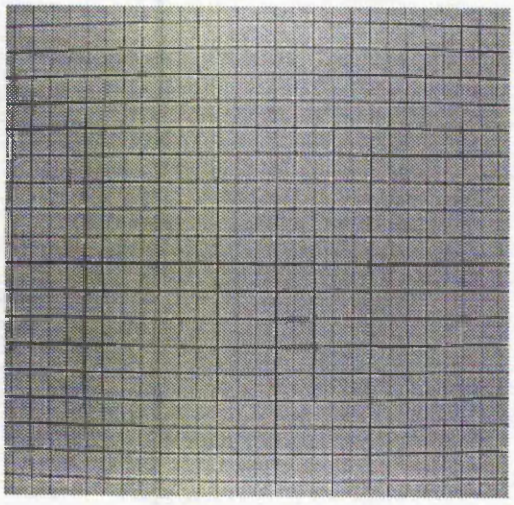


Figure 2-14: *A grid pattern captured by a CCD camera*

view the part "straight on" in the middle of the field and at a relatively large angle at the edge. If using focal length lenses results in a more constant viewing angle and small magnification errors over the depth of field, how long must a lens focal length be and how far away from the object must the camera be to produce "zero" viewing angle and magnification (the ratio of the size of the image to the size of the object or part) error ? A very long distance indeed - infinitely far away.

2.5.1 Correcting by software filter

The accurate imaging of relatively flat objects, where the object thickness is very small (say 1%) compared to the object's length and width, is a much easier task than is the imaging of 3-dimensional objects.

In this project only flat objects, such as paper and lace web are used. A simple software filter is developed to solve the view angle error caused by the fish eye lens. For instance, a vision system is organised as 256 by 256 pixels and mapped into frame memory. By analysing the grid image shown in Figure 2-14, it is possible to find suitable correcting ratios in both abscissa and ordinate. A 256 by 256 array filter is built to record these measured correcting ratios which can be used to recover the distorted image.

2.5.2 Correcting by telecentric lens

For conventional camera lenses, the viewing angle increasingly tilts inward across the field and at the edge may be tilted by 'tens of degrees', depending on the focal length of the lens and the size of the field of view. These changes in the viewing angle can cause real problems in machine vision applications.

Facing this problem, design engineers have adopted a technique of presenting a part from a number of different views, with each view detailing what we would see if looking in a

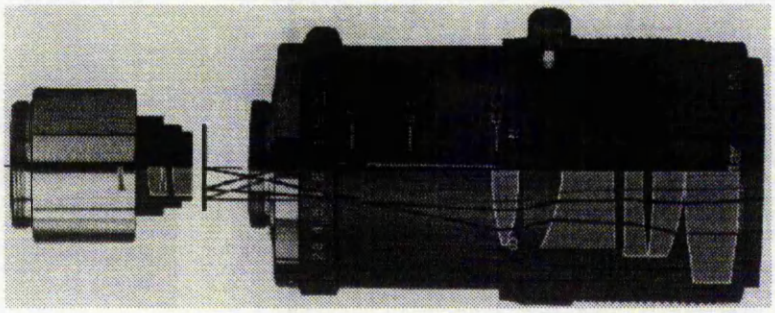


Figure 2-15: *Telecentric lens and adaptor*

given direction. Within a single view, all lines of sight are parallel. This is, within the view, we do not vary the viewing angle. This is a crucial concept because if engineers want to inspect the part in a manner consistent with the drawing, engineers want the imaging lens and inspection system to also maintain a constant viewing angle across the entire view.

Optical engineers describe lenses which maintain a constant viewing angle as being "telecentric", where errors in telecentricity are usually a few degrees at most. For example, a lens which is telecentric to within 1.0 degree has a viewing angle of 1.0 degree at the edge of the field of view compared with the viewing angle in the middle of the filed, which is usually defined as 0.0 degree.

In contrast, the 'normal' lens on a 8.5 mm camera has more than a 40 degree viewing angle change between the center and edge of its field. Telecentric lenses (Figure 2-15), designed especially for machine vision applications, are now available which have very close to ideal "zero" viewing error while keeping the camera to part distance consistent with inspection equipment size limitations. Compared to conventional camera-type lenses, telecentric lenses reduce viewing angle and magnification errors by a factor of 10 or more. (Refer to Appendix A for the detailed information on this matter)

2.5.3 Implementation

In this project, we have employed the software filter to recover the distorted image (for cost effectiveness). It is noted that in order to speed up this operation a simplified software recovering method has been used. As already mentioned, when a conventional camera lens is used to inspect an object, the lens views the part 'straight on' in the middle of the field and at a relatively large angle at the edge. If the one can manage the vision system only using the middle (centre) part of the field to view an object, most of the viewing angle error can be ignored. Engineers can simply calculate the magnification error and find a proper correcting ratios in both X and Y axes. The pair of correcting ratios can be used to recover the distorted image through the entire object.

Figure 2-16 shows two processed sample patterns. As depicted, the original image contains 256 by 256 pixels. However, the actual size of the image analysed is only 226 by 226 pixels around the middle of the field (from location (15, 15) to (240, 240), depicted in Figure 2-16).

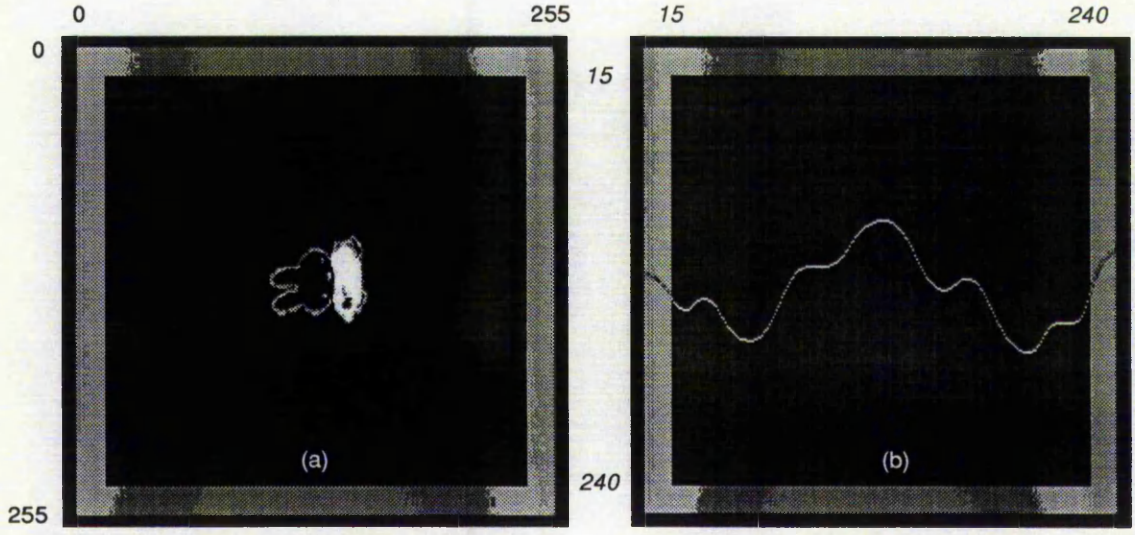


Figure 2-16: *Two processed sample patterns*

(a) (b)

Figure 2-17: *Corrected patterns (1.45862 : 1)*

Assuming the camera used a 8.5 mm conventional lens, the correcting ratios (X = 1.45862 and Y = 1) can be obtained. The pair of correcting ratios are multiplied by the vectors of the extracted objects. The binary patterns can be converted as shown in Figure 2-17. However, by using this approach a small amount of viewing angle error will still appear within the converted image. For instance, when a pattern is captured (8.5 mm lens) at the working distance (object to lens) of 35 cm or 13.78 inch, according to the experimental results, the maximum error detected is approximately 1 mm in both X and Y axis directions. However, in the project, this inaccuracy can still be tolerated. Indeed, if a machine vision application is highly dependent on precision measurement, it is highly recommend to employ the telecentric lenses rather than using software filters.

Figure 2-17 shows the corrected patterns using the simplified software converting method. Take an example of case (b), only 226 vectors (data points) are used to record the path (from X_{15} to X_{240}). The detected vectors are transformed into a set of machine movement data and down-loaded into the controller of the cutting mechanism.

2.6 Summary

This chapter discusses the vision system and its relevant components employed in the project. It also briefly introduces the techniques used to extract a binary pattern from a gray-scale image. The methods applied to correct a distorted image are also presented.

Particular attention is paid to the thresholding operation which is used for bi-leveling a gray-scale image into a binary bitmap. Two methods have been described. The first method is based on analysing the light intensity histogram (optimal thresholding). The second technique, developed within the project, is to find the average intensity of each vertical strip of the image and examine which points lie above and below this threshold. According to numerous experimental results applying the *average intensity analysis* has a better outcome than using the conventional histogram analysis.

As a pattern is extracted from an image, after line skeleton operation or border following process, the cutting path is created and fitted into a software filter where the distorted pattern can be converted. A simplified software recovering method has been presented to speed up this correcting process. An alternative approach using telecentric lenses to correct the distortion has been described.

3. MACHINE VISION AND CONTROL

Chapter 3
MACHINE VISION AND CONTROL

3.1 Introduction

A CNC cutting machine (Pacer Compact 800+) is used together with the vision system as the testing rig in the project. The CNC machine is controlled by an enhanced HPGL language via a *DOS device drive* or via a *serial interface*. By setting a device drive (PACER$) in CONFIG.SYS file, which allows engineers to configure their own developed environments and adjust the control commands, selecting operating parameters by a set of commands from the host system.

There are two kinds of machine controller (68k processor based) available for use: a) VMEbus based and b) PC based. The PC based controller, which is used in the project, is made of one printed circuit board that is installed within the PC or via RS232 connected to an IBM compatible PC which acts as a machine console. The loading of the timer registers occurs at the end of each interpolation movement. The rate at which the controller accesses the timers is proportional to the speed of motion. The axes positions are only updated at the end of each interpolation movement [MAC92].

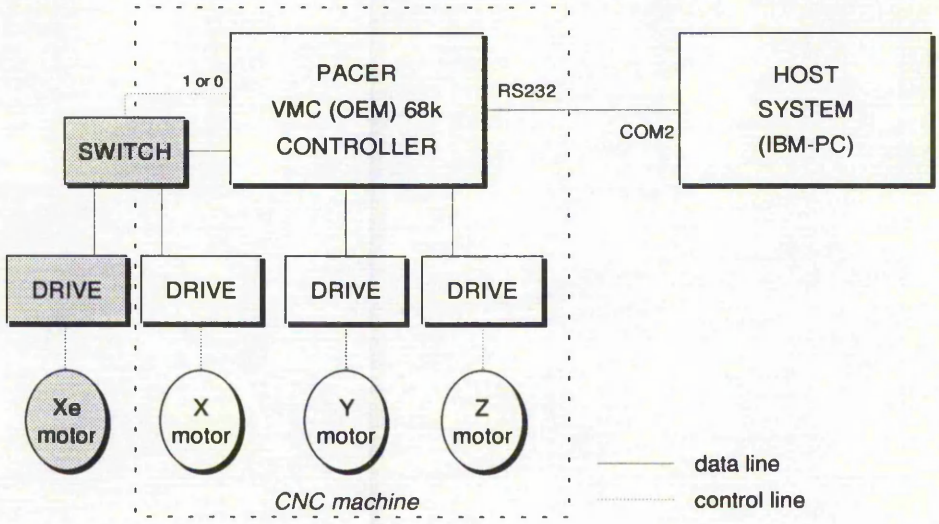


Figure 3-1: *Block diagram of a PC based CNC system*

3.2 System Configuration

As depicted in Figure 3-1, motors are connected to a standard three-axes VMC controller. An extra relay device is used to switch data signals between two motors. This switch is triggered by an external signal from the host system. In the test rig, the device is used to swap control commands between X axis motor and Xe (driving a conveyor system) motor. A similar approach can be utilised to attach more extra motors to the CNC system.

There are two ways in which a host system can communicate with the VMC controller. The first method is to write a set of HPGL commands and copy them into the device drive. The second approach is to use a serial link controlled by Kermit directly sending the command to the controller. MS-Kermit is a program that implements the *Kermit file transfer protocol* for the entire IBM PC family. It performs two major functions, terminal emulation and file transfer. Both approaches have been used during the development. In the final version of the programs, only the first method is used (refer to Appendix B for more information about the PACER VMC HPGL).

3.3 Calibration of the Camera

It is essential to properly register (calibrate) the camera with the CNC machine before the vision system delivers the control information for the cutting process. Two schemes have been proposed for the registration of the machine vision system: 1) Manual calibration, and 2) Automatic calibration.

3.3.1 Manual calibration

As illustrated in Figure 3-2, in the control machine system the camera is used to guide the machine based on the analysed pattern. The pattern is captured by the camera (8.5 mm lens) at the working distance of 35 cm (13.78 inch). A white paper marked with a black cross is placed underneath the camera. The image is analysed, the location of the crossing point is calculated and passed to the cutting mechanism. This information is then used to guide the cutter (pen) to the target position (centre of the cross).

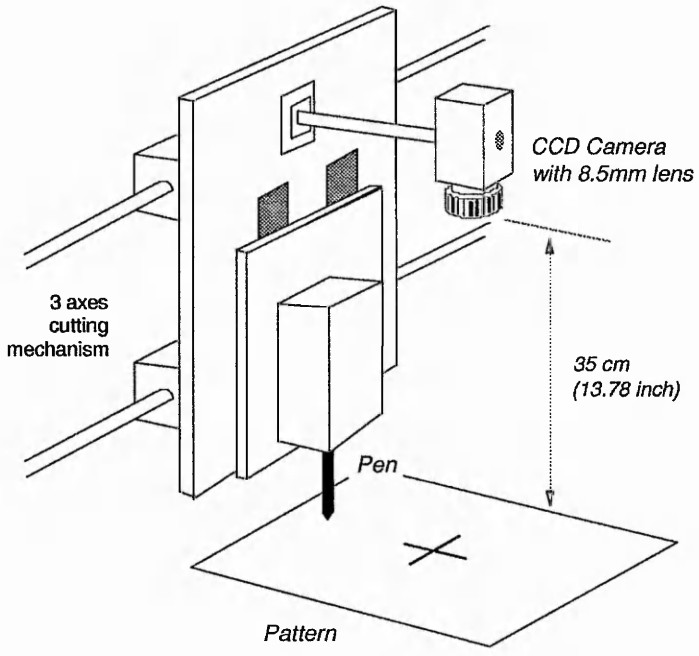


Figure 3-2: *Schematic of the machine vision system*

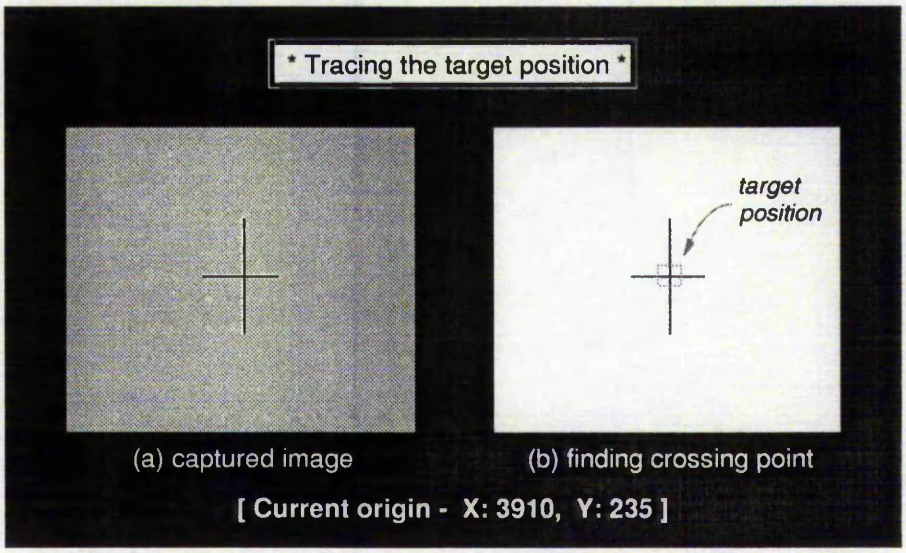


Figure 3-3: *Processes of tracing the crossing point*

By visual inspection, if the current calibration is inaccurate a gap (mismatch) is found between the pen and the target position. Measuring the distances of the gap in X and Y axis direction, a suitable modification can be determined. A configuration file has been created to record all the system parameters. Simply modifying the related data (ASCII format) within the file, the updated configuration will adjust the parameters as soon as the system is re-started. This method is based on a trial and error approach which is depending on the experience of the operator. Figure 3-3 shows the main stages of detecting the target position (crossing point).

3.3.2 Automatic calibration

Instead of using the method described above, an alternative approach is to program the system to automatically calibrate itself. To achieve this, first, the machine is set to draw a cross on white paper. Then, move the machine back to its origin and capture a frame of the drawing where the location of cross point is found. Depending on the relative position of this cross point, the inaccuracy can be measured and corrected.

Unfortunately, two difficulties have been found using this scheme. Firstly, the resolution of the vision system is about 1 mm per pixel, which means that the system cannot detect any error (gap) smaller than this value. Secondly, a pen has to be mounted on the cutting head of the CNC machine. In the other words, if a cutter is employed in the system, it is not possible to draw a cross on paper for the calibration. However, for the commercial type of machine, the first problem mentioned above can be overcome by means of adapting a higher resolution vision system. Also, a better *camera mounting bracket* can be engaged to ensure the steadied mounting with the cutting mechanism.

3.4 Path Following Process

When a pattern is captured by the vision system and analysed, a set of control commands is created and sent to the cutting mechanism. Basically speaking, no matter what the shape of the pattern is, the cutting path created is composed of a series of path following commands. The machine follows a batch of movement data which can be a straight line, an arc, or a circle. Depending on the size of the movement data in a batch, the average time taken to pre-process these data into control commands is approximately 1-4 seconds in the 68k based machine controller.

Figure 3-4 illustrates two samples collected from the path following process. A second line is drawn by the testing rig with a felt-tip pen connected on the Z axis to replace a cutter. As already mentioned in Section 2.5, since a conventional vision system is employed in the project, the fish-eye lens of the camera can create view angle and magnification errors. A simplified software filter (Section 2.5.3) is developed for rapidly correcting the distorted image. However, a small amount of view angle error will still appear within the converted image. Once the converted image is analysed to produce a set of vectors for controlling the cutting mechanism, the result of this path following process will not be completely accurate

(carefully inspect the samples in Figure 3-4). Nevertheless, this inaccuracy can still be tolerated throughout the process.

Furthermore, in order to increase the speed of this transformation, as the batch is detected larger than a certain amount of data, these data are cut into small pieces and transmitted to the machine controller. By adopting this approach, a large amount of time can be saved by waiting for the controller to process the data. Only vectors are used to record the detected cutting path, which is a set of polylines. Of course it is possible to generate a smoother path by using circular arcs instead of employing straight lines, but this is not the objective of the project.

3.5 Connection between Frames

As shown in Figure 3-2, a CCD camera is mounted on a three-axes cutting mechanism. The camera is moved with the cutting head in both X and Y axis direction (the camera is not connected with the Z axis). This keeps the camera and the cutter in a constant position relative to each other. Consequently, it is easy to detect the position of the object from the

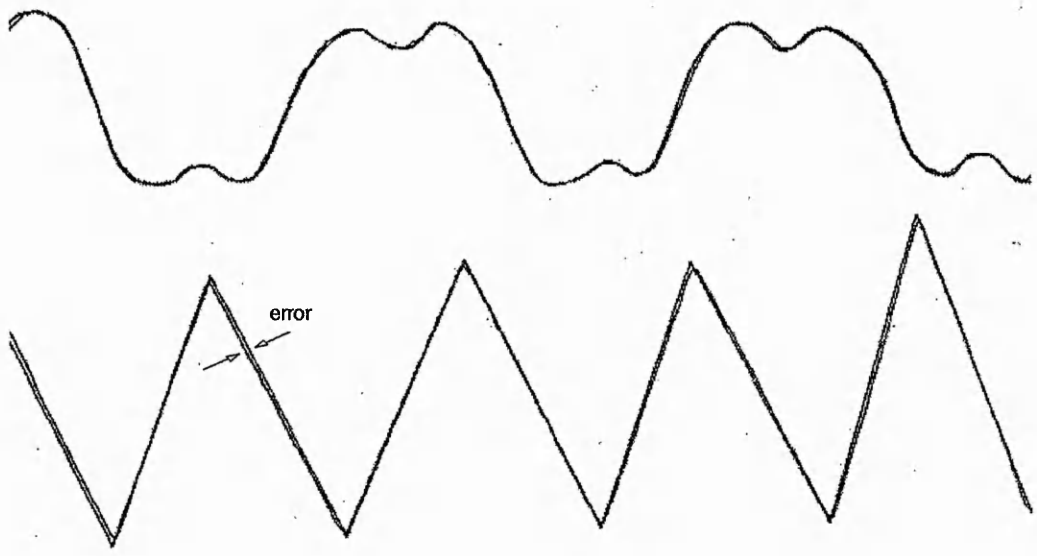


Figure 3-4: *Two samples of path following process*

captured image, and to correct the errors between frames. Besides, since the camera is followed by the X and Y axes of the machine, the camera can capture images at different locations of the workplace. This enables the wider viewing field of the vision system. The camera can be set to have lower working distance (object to lens) where the resolution of the camera is higher.

The vision system captures an image while the machine is cutting / drawing an object. The vision system has to consider the more complex two dimensional image shifting problem. However, this approach yields more accurate results than the conventional setting, where the camera is fixed instead of rowing around with the cutter. This method also avoids small drifts due to mechanical slippage. The method for analysing shifting images in two directions and determining a suitable position for capturing the next frame are described next.

3.5.1 Detection of the capture point

When the first frame of the pattern is captured and bi-leveled, the centre line between the upper and lower boundaries of the pattern in the frame is found. The last (most right hand side) intersection between the analysed pattern and the centre line is located (Figure 3-5). This position is used for capturing the next frame of the pattern which is designated as the 'capture point'.

The machine starts moving from the 'start of cutting' position and stops at the 'end of cutting' position, which are labelled in Figure 3-5. As the machine reaches the capture point within the path during the cutting process, the vision system is triggered to take a new image frame which is temporarily stored in the memory.

Besides, the machine continuously moves until it reaches the 'end of cutting' point. The distance ('L' indicated in Figure 3-5) between the *capture* and the *end of cutting points* is the

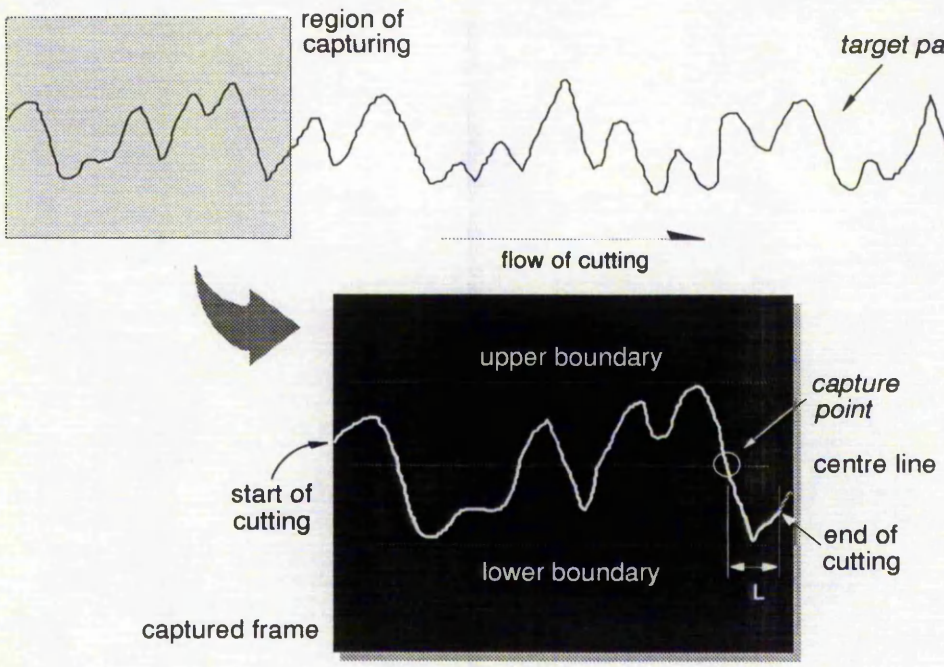


Figure 3-5: *Example of detecting the capture point between frames*

key where the cutting machine can continuously work between frames - Since the process of capturing and analysing a frame normally takes the CPU a few hundred milli-seconds, we add an extra length (say 30 vectors) of the path behind the capture point (see Figure 3-5). Before the machine completely ends the cutting process, a new batch of the vectors from the second captured image has already been obtained and down loaded to the machine controller. This enables the cutting process to operate in real-time.

Figure 3-6 represents the calculation of the cutting path and capture point within the second captured frame. Indeed, this approach not only can deal with the pattern which has small deviation from a straight line (see Figure 3-6), but also can be used to analyse a curve which has large deviation from the straight line, such as illustrated in Figure 3-7.

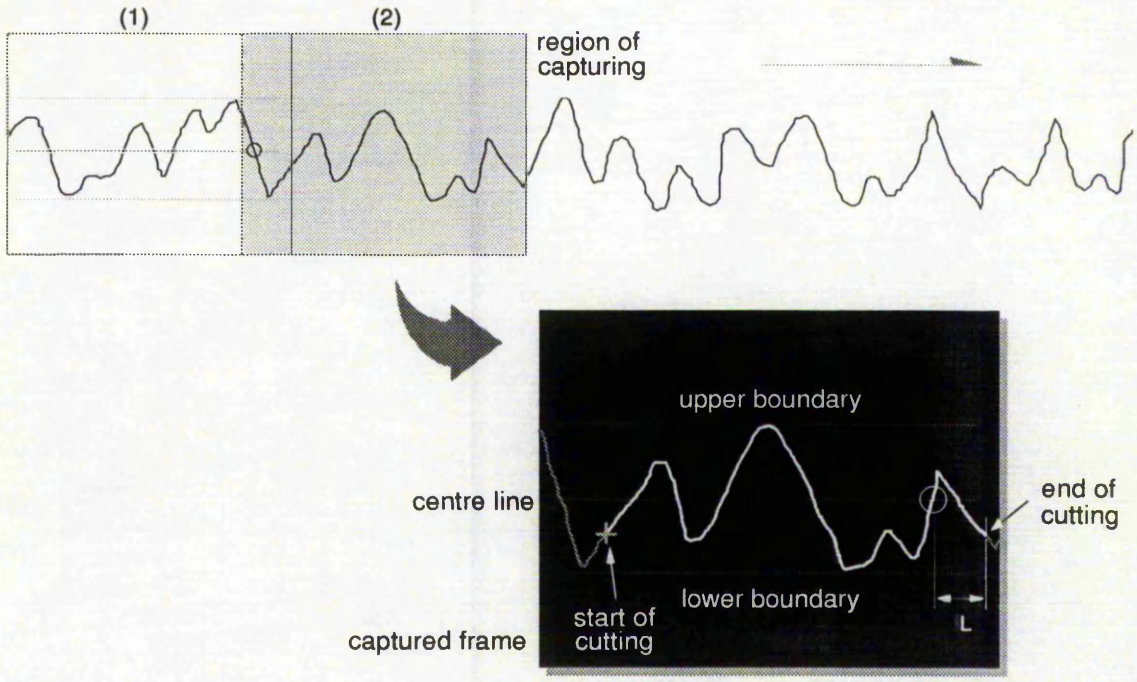


Figure 3-6: *Detecting the cutting path and the capture point within the second frame*

3.5.2 Calculating the location of the detected path

Once the cutting path is extracted from a captured image, the location of the path has to be transformed into an absolute position related to the bed of the machine. As already stated in Section 2.5.3, a camera with 8.5 mm conventional lens has the correcting ratio of (X: 1.45862, Y: 1). Also the camera is installed with the working distance of 35 cm (see Figure 3-2). According to the experimental results, the magnification (the ratio of the size of the image to the size of the object or part) is 29. Hence, the actual parameters applied to the image are (X: 42.3, Y: 29.0). The following Equations represent the computation of the process.

$$Xn = O_X + (V_{Xn} \times P_X)$$

$$Yn = O_Y + (V_{Yn} \times P_Y), \quad n = 15 \ldots 240$$

where Xn and Yn are the transformed data;

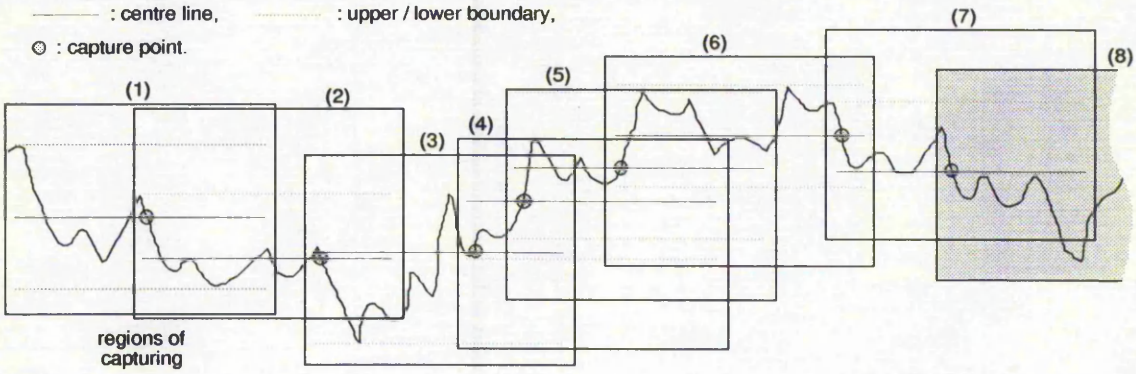


Figure 3-7: *Example of capturing the images between frames*

O_X and O_Y are the position when the camera captures an image (origin);

V_{Xn} and V_{Yn} are the original vectors;

P_X and P_Y are the actual correcting parameters (42.3, 29.0); and

'n' is the n^{th} vector (data).

Using this approach, the absolute position of the cutting path can be obtained. The host computer triggers the vision system to grab an image. It sends a request to the VMC controller in which the controller will reply with the current cutting head position to the host computer (O_X , O_Y). Since the Frame Grabber digitises an incoming video signal at a rate of 33 mill-seconds per frame, this delay will directly affect the accuracy of the obtained position data (position of the cutter). In other words, the image-capturing process has actually happened 'after' the host PC receives the position data. Figure 3-8 shows a possible hierarchy of this event.

To avoid this inaccuracy, the relationship between the moving speed and the direction of movement must be calculated while the machine is capturing an image. A reasonable correction can be obtained to compensate this error.

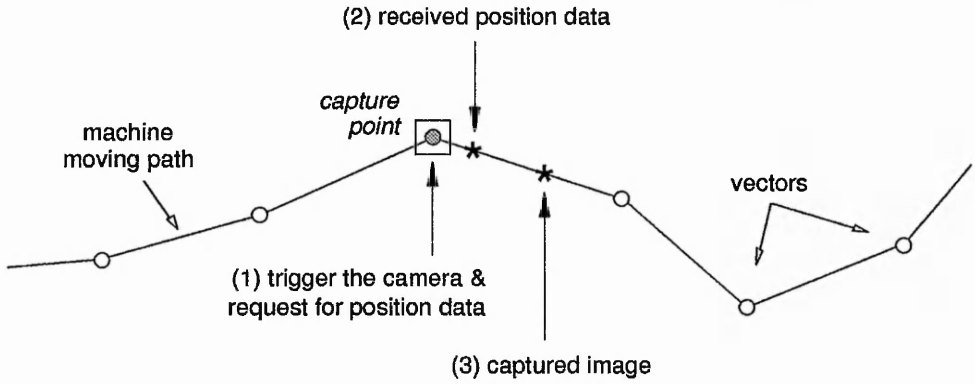


Figure 3-8: *Events that happen when capturing an image and asking for the position data*

3.6 Summary

This discusses the cutting mechanism and the machine vision process developed in the project. The selected 68k based three-axis CNC cutting machine is described. In order to construct a conveyor system on the cutting machine, an extra motor has been attached on the VMC controller.

The methods developed for calibrating the vision system with the cutting mechanism have been presented. The advantages and disadvantages of applying either approaches, i.e., manual calibration or automatic calibration, are discussed. A scheme for managing the control system to continuously grab new images as well as guiding the machine to trace the targets in real-time is developed. Finally, an equation has been derived and used to transform the detected vectors into absolute position on the bed of the machine.

4. LACE PATTERN DETECTION

Chapter 4
LACE PATTERN DETECTION

4.1 Introduction

Lace manufacturing is an important industry worldwide, and particularly for the East Midlands in the United Kingdom. Lace is commonly knitted on Raschel machines which cost up to £500,000 each and must run 24 hours per day for profitable operation. Typically four such machines are supervised by one operator, who patrols and attempts to spot any defects visually, as soon as it occurs.

Lace is manufactured in webs up to 3.5 meters wide with many pattern repeats across the width. It is supplied to garment manufacturers in long lengths wound on a reel and must be cut into short strips before sewing. For reasons of appearance the lace must be cut into two matching mirror image pieces so that the finished trim will be symmetrical about the centre line of the garment. The point where the lace pattern is cut must also be consistent so that all garments from a production batch have a similar appearance. It is essential therefore that the separation is a high quality operation so as not to impede sales of the finished material.

The preparation of the short lengths of lace trim for sewing onto garments is typical of many processes in the clothing industry - highly labour intensive and completely lacking in automation. At the present time lace is cut to size in the following manner. Lengths of lace several meters long are impaled on a row of nails. Successive layers of lace are placed on top of each other in mirror image pairs with care being taken to line up the pattern correctly. When approximately fifty layers have been built up a sharp knife is used to cut the pile into short sections with each cut at a predetermined position in the lace design. The result is identical piles of lace pieces arranged in matched mirror image pairs.

Presently the scalloping of lace is performed manually, which is a lengthy and expensive process and results in slowing the rate of production. A small number of machines have been developed that use a simple passive cutter mechanism that relies on the structural strength of the lace pattern. In such systems the cutter is held stationary while the lace is run against it. This approach is only suitable in cases where the lace pattern is of very shallow scallop. In the case of deep scallop patterns a more sophisticated method of guidance, employing an active cutter, is required.

Although modern lace production methods are mechanised there is still considerable variation between samples of the same lace even when taken from the same production batch. Unlike rigid engineering materials lace has essentially no stiffness and can stretch, shrink, wrinkle and distort in other ways depending on the conditions under which it has been stored and its state of tension when viewed. For this reason some form of sensing is required to ensure that succeeding lengths of lace are cut at the same point in the lace design.

Cutting or 'scalloping' lace might seem a million miles from the traditional concerns of instrumentation and control. In fact, and incremental, vision based machine control systems using CCD cameras have potential for all kinds of web applications where deformable materials are processed. Machine vision is the most promising methodology for meeting

Figure 4-1: *A typical lace pattern*

these requirements. Unfortunately, lace comprises a fine and complex pattern of threads (Figure 4-1), which must be identified in real time. Even immediately below the point of knitting, small distortions are unavoidable; the lace rapidly contracts inwards at the edges by several centimetres. Moreover, the machine oscillates at about 6 Hz. close to the point of knitting; the lace moves both backwards and forwards at each cycle, with a net advance of about 1 millimetre. Thus the lace advances at no more than 6 mm/second. These facts present a number of problems to the typical image processing setup [SAN95].

An experimental vision based system has been developed (Figure 4-2) for automation lace scalloping. On-line pattern recognition is performed to detect the cutting path (river), which is indicated in Figure 4-1. The cutting path is vectorised and transferred to a trimming mechanism. In order to satisfy industrial requirements two main conditions must be satisfied. To achieve a sufficient degree of automation: Firstly, the river must be found without prior knowledge of the lace pattern scanned. Two attempts which applied *traditional image processing methods* and *fuzzy reasoning rule-based technique* have been made to detect the river within various lace patterns. Secondly, finding the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach called the *Line Mapping Method (LMM)* is used to speed up the search for the river in subsequent frames.

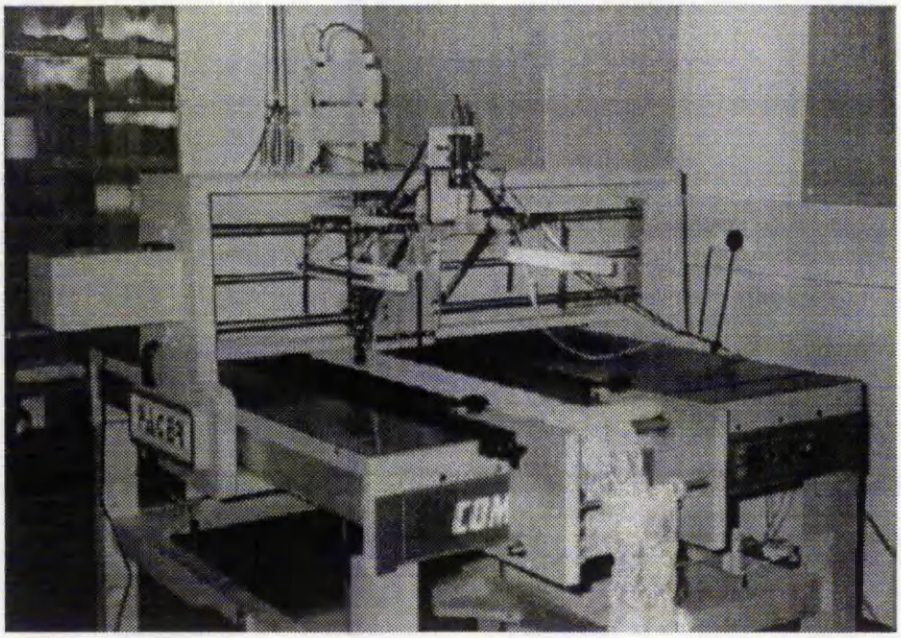


Figure 4-2: *Prototype of a vision based lace trimming system*

A bi-level image, shown in Figure 4-3, is used. After a thresholding operation a river shows up as a dark area (pixel group) within the edges that crosses from one side of the image to the other in a nearly unbroken sequence. There are *thick threads* that cross the river at intervals. These are indistinguishable from the material surrounding the river (marked by circles in Figure 4-3). Allowance must be made for small breaks in continuity of the river due to these cross threads.

The approaches developed can be structured into three sections: *detection of first cutting river*, *line mapping process* and *supervision of the system.* The vision based lace trimming system is performed using an IBM PC and a video frame grabber with a CCD array scan camera as input device. Image are displayed on a monochrome video monitor. The vision system has the standard resolution of maximum 256 pixels by 256, 8 bits (256 gray levels) per pixel. All software was written in "C". In the following section the author presents the concept of the *fuzzy logic* algorithm employed in the system for detection of the first cutting river across the lace pattern.

Figure 4-3: *Bi-level lace bitmap (binary) image*

4.2 Fuzzy Logic

During the past few years, fuzzy logic has been finding a rapidly growing number of applications in fields ranging from consumer electronics and photography to medical diagnosis systems and securities management funds. What is exploited in most of these applications is the tolerance for imprecision.

In contrast to classical logical systems, fuzzy logic is aimed at a formalisation of modes of reasoning which are approximate rather than exact. Basically, a fuzzy logical system may be viewed as a result of fuzzifying a standard logical system. Thus, one may speak of fuzzy predicate logic, fuzzy modal logic, fuzzy default logic, fuzzy multi-valued logic, fuzzy epistemic logic, etc. In this perspective, fuzzy logic is essentially a union of fuzzified logical systems, and precise reasoning may be viewed as a special case of approximate reasoning.

4.2.1 Introduction

Fuzzy logic is a vigorous, yet forthright, problem solving technique with wide-ranging applicability [VIO93]. In general, it is most useful in handling problems not easily definable

by practical mathematical models, such as lace pattern detection, etc.. Fuzzy logic derives much of its power from its ability to draw conclusions and generate responses based on vague, ambiguous, qualitative, incomplete, or imprecise information. In this respect, fuzzy-based systems have a reasoning ability similar to that of humans [VIO93].

Lotfi Zadeh is most widely associated with fuzzy logic [ZAD88]. In 1965 he presented the original paper formally defining fuzzy set theory, from which fuzzy logic emerged. Zadeh extended traditional theory to solve the paradoxes sometimes generated from the "nothing-or-all" classifications of Aristotelian logic. Traditionally, a logic premise has two extremes: either entirely true or entirely false. However in the fuzzy world, a premise ranges in degree of truth from 0 to 100 percent, which allows it to be partially true and partially false.

Because fuzzy logic is rule based, it only requires a small amount of memory and it is nonlinear, meaning that it can perform very complicated computations more easily than traditional methods. The goal of fuzzy logic is to simplify a complicated application and eliminate expensive tools, complex language, processors, and massive amounts of mathematical logic.

Figure 4-4 illustrates the flow of data through a fuzzy system [VIO93]. System inputs undergo three transformations to become system outputs. First, a fuzzification process that uses pre-defined membership functions maps each system input into one or more degrees of membership. Then, the rules in the rule base are evaluated by combining degrees of membership to form output strengths. And lastly, the defuzzification process computes system outputs based on strengths and membership functions.

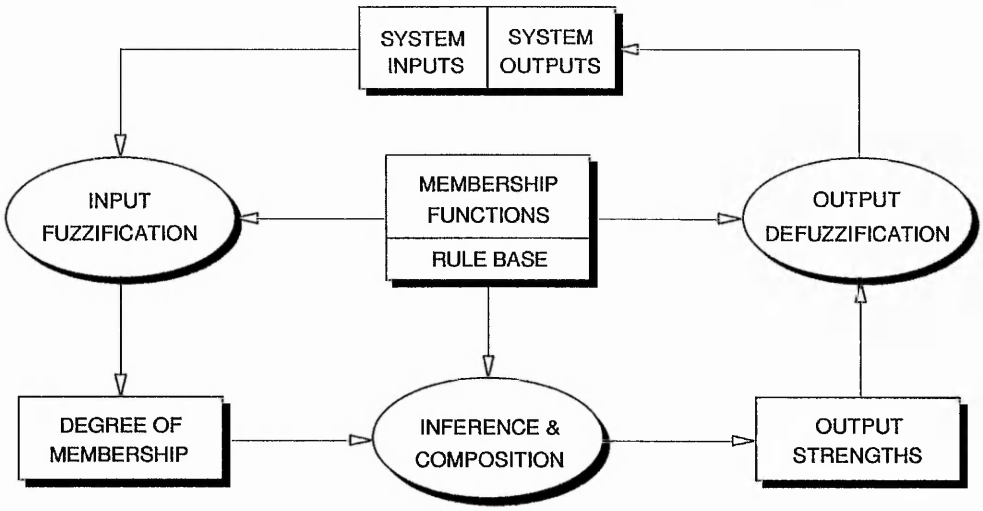


Figure 4-4: *Data flow of a fuzzy-logic-based system*

4.2.2 Principles

Fuzzy logic provides a wide variety of concepts and techniques for representing and inferring from knowledge which is imprecise, uncertain or lacking in reliability. At this juncture, however, what is used in most practical applications is a relatively restricted and yet important part of fuzzy logic centering on the use of fuzzy if-then rules. This part of fuzzy logic will be referred to as the *calculus of fuzzy if-then rules* [ZAD92] because it constitutes a fairly self-contained collection of concepts and methods for dealing with varieties of knowledge which can be represented in the form of a finite number of if-then rules in which the antecedents and/or consequents are fuzzy rather than crisp. The importance of the calculus of fuzzy if-then rules stems from the fact that much of human knowledge lends itself to representation in the form of a hierarchy of fuzzy if-then rules. Furthermore, the calculus of fuzzy if-then rules provides an effective method for dealing with man/machine systems in a qualitative framework.

Mathematical equations that describe the calculus of fuzzy if-then rules are presented here. They are presented without derivations or proofs. This information can be found in Jamshidi [JAM93], Lee [LEE90] and Zadeh [ZAD85/88/92]. The point of departure in the

calculus of fuzzy if-then rules is the interpretation of a fuzzy if-then rule. If we assume for simplicity that the antecedent and consequent of a rule have a single conjunct, the rule may be expressed as

$$\text{if } X \text{ is } A \text{ then } Y \text{ is } B$$

where B and A are fuzzy predicates in U and V which play the role of elastic constraints on X and Y. In terms of these constraints, the rule may be interpreted in two ways. First, as a joint possibility distribution of X and Y, in which case the possibility distribution in question may be expressed as

$$\Pi (X,Y) = A \times B$$

where $A \times B$ represents the Cartesian product of A and B. This implies that the possibility distribution function of X and Y may be expressed as

$$\pi (u,v) = \mu_A(u) \wedge \mu_B(v)$$

where μ_A and μ_B are the membership functions of A and B, respectively, and $\wedge$ denotes the 'minimum' operator. In the second interpretation, the rule in question is interpreted as a conditional possibility distribution, $\Pi (X \mid Y)$, of Y given X. More specifically,

$$\Pi (X \mid Y) = \neg A \oplus B$$

where '$\neg A$' is the negation of A and $\oplus$ is the bounded sum.

A basic question is : How should two or more rules be combined ? In the case of the joint possibility distribution, the interpretations are combined disjunctively. More specifically, a collection of fuzzy if-then rules of the form

$$\text{if } X \text{ is } A_i \text{ then } Y \text{ is } B_i\,,\ i = 1, \cdots, n$$

defines the joint possibility distribution

$$\Pi\,(X,Y) = A_1 \times B_1 + \cdots + A_n \times B_n$$

in which '+' plays the role of disjunction. Thus, if the Cartesian product $A_i \times B_i$ is visualised as a fuzzy point, $\Pi\,(X,Y)$ may be viewed as a fuzzy graph or fuzzy relation R which may be expressed as

$$R = \sum_i (A_i \times B_i)$$

If a rule is interpreted as a conditional possibility distribution, the membership function of the output may be expressed as

$$\mu_B(v) = sup_u(1 \wedge (\wedge_i (1 - \mu_{A_i}(u) + \mu_{B_i}(v))) \wedge \mu_{A_i}(u)$$

where the *supremum* is taken over the domain of X. It can be shown that if b_i is taken to be the output of rule i when input X is A is applied, then B is contained in the intersection of the b_i, i.e.,

$$B \subset b_1 \cap b_2 \cap \ldots \cap b_n$$

From the entailment principle of fuzzy logic, it follows that the output may be expressed as

$$Y \text{ is } (b_1 \cap \ldots \cap b_n)$$

In qualitative systems analysis and intelligent control, a fuzzy rule set may be interpreted as a qualitative description of the input-output relation of a system. Thus, a rule set of the form

$$\begin{array}{l} \textit{if } X \textit{ is } A_1 \textit{ then } Y \textit{ is } B_1 \\ \textit{if } X \textit{ is } A_2 \textit{ then } Y \textit{ is } B_2 \\ \hline \textit{if } X \textit{ is } A_n \textit{ then } Y \textit{ is } B_n \end{array}$$

may represent the input-output relation of a system R, which the relation in question expressed as the fuzzy relation

$$R(X,Y) = \sum_i (A_i \times B_i)$$

In the case of a serial combination of *R1* and *R2*, the problem is to derive the input-output relation of the serial combination, *R12*, from the knowledge of the input-output relation *R1* and *R2*. In the case of *R1*, the fuzzy if-then rules are of the form

$$\textit{if } (X_1 \textit{ is } A_1^1 \textit{ and } X_2 \textit{ is } A_1^2) \textit{ then } (Y_1 \textit{ is } B_1^1 \textit{ and } Y_2 \textit{ is } B_1^2)$$

and the corresponding fuzzy input-output relation may be expressed as

$$R1(X_1, X_2, Y_1, Y_2) = \sum_i (A_i^1 \times A_i^2 \times B_i^1 \times B_i^2)$$

The same approach may be used to compute the input-output relation of any combination of systems each of which is characterised by a collection of fuzzy if-then rules. This makes it possible to analyse in qualitative terms the behaviour of a complex system whose constituents are characterised by fuzzy rule-sets with linguistic variables rather than algebraic or differential equations. Important application areas for this approach are industrial process control, prediction and signal processing, and large scale system modeling.

Another important application area relates to probabilistic systems in which the underlying probabilities are not know with sufficient precision to be treated as numbers. The calculus of fuzzy if-then rules provides a basis for dealing with such probabilities as collections of fuzzy if-then rules. More specifically, assume that X and Y are random variables whose probability distributions are described in linguistic terms. For instance,

$p(X)$: if X is *small* then probability is *low*
if X is *medium* then probability is *high*
if X is *large* then probability is *low*
.....

$q(Y \mid X)$: if (X is *small* and Y is *large*) then probability is *high*
.....

where $q(Y \mid X)$ is the conditional probability distribution of Y given X. The problem is to compute the probability distribution of Y in the form of a collection of fuzzy if-then rules with linguistic variables.

Since computation with probabilities involves the operations of addition and multiplication, it is necessary to have a method for operating on functions characterised by collections of fuzzy if-then rules. For example, if $f(X)$ and $g(Y)$ are described in terms of such rules as disjunctions of fuzzy points then $f(X)$ and $g(Y)$ may be expressed as

$$f(X) = \sum_i (F_i \times F_i')$$

and

$$g(Y) = \sum_j (G_j \times G_j')$$

Furthermore, if $f(X)$ and g(Y) are combined through a binary operation '*', e.g., multiplication or addition, then the resulting function may be expressed as

$$h(X,Y) = f(X) * g(Y)$$

It can be shown that the fuzzy rule set associated with $h(X,Y)$ is given by

$$h(X,Y) = \sum_{i,j} \{(F_i \times G_j) \times (F_i' * G_j')\}$$

This result provides the basis for computation for imprecisely-known probabilities which are expressed as collections of fuzzy if-then rules involving linguistic variables. In the case of such probabilities, we have

$$p(X) = \sum_i (P_i \times P_i')$$
$$q(Y \mid X) = \sum_j (Q_j \times Q_j')$$
$$r(X,Y) = p(X)\, q(Y \mid X)$$

In the application of the calculus of fuzzy if-then rules, a problem of basic importance relates to the calibration of the membership functions of the linguistic values. The current practice in the design of fuzzy-logic-based systems employing such rules involves, for the most part, the use of cut-and-trial procedures. Recently, however, researches have been taken towards the development of adjustment algorithms - some of which are based on neural network techniques - for a systematic approach to the problem of calibration.

The calculus of fuzzy if-then rules provides a systematic way of dealing with systems in which it is either necessary or advantageous to describe the input-output relations in the form of fuzzy if-then rules. The calculus of fuzzy if-then rules is simple and close to intuition. Furthermore, it is largely self-contained and does not require an extensive familiarity with fuzzy logic. For these reasons, the calculus of fuzzy if-then rules has become a widely used tool in systems analysis, control, pattern recognition, decision analysis, diagnostics and related fields [ZAD92].

4.2.3 Implementation

A fuzzy algorithm is an ordered sequence of instructions that may contain fuzzy assignment and conditional statements. In practice, engineers can express the fuzzy conditional statements in the form:

IF *A* THEN *B*

where *A* and *B* have fuzzy meaning, for instance

IF *x is centre* THEN *y is large*

where *centre* and *large* are viewed as labels of fuzzy sets.

Degrees of truth are the major currency of production systems. Production systems are based on repetitive use of the construct

IF [antecedent] THEN [consequence]

This construct is applicable to situations in which there is usually only one correct decision. Because we cannot be entirely certain that some facts are true or that certain relations hold, each fact and each production rule are associated with a certainty factor. The certainty factor, a number in the interval [0, 100](%), indicates the certainty with which each fact or rule is believed. Inexact reasoning is based on the construct

IF [antecedents (*to degree* X_n)]
THEN [consequences (*to degree* Y_n)]

Such constructs are applicable to decisions involving phenomenological data and situations in which there might be more than one correct decision. For example, consider these two rules, with respective certainty factors of 90(%) and 80(%), leading to conclusion $D_{1\&2}$:

IF [A and B and C] THEN [$D_1 = 0.9$]

IF [E and F and G] THEN [$D_2 = 0.8$]

Additionally, suppose that the facts A, B, C, D, E, F, and G have certainty factors 0.6, 0.3, 0.5, 0.8, 0.6, and 0.7, respectively. Then the following computation produces a certainty factor of 0.48 for system output D (applying *min-max inference*):

IF $\quad min\ (A, B, C)^{certainty\ factors} = min\ (0.6, 0.3, 0.5) = 0.3$

THEN $\quad 0.3 \times 0.9 = 0.27\ (degree\ of\ output, D_1)$

IF $\quad min\ (E, F, G)^{certainty\ factors} = min\ (0.8, 0.6, 0.7) = 0.6$

THEN $\quad 0.6 \times 0.8 = 0.48\ (degree\ of\ output, D_2)$

Therefore, the combination degree of conclusion D is

$$max\ (D_1, D_2)^{\ degree\ of\ output} = max\ (0.27, 0.48) = 0.48$$

By combining this degree of truth concept, fuzzy logic extends conventional logic in two ways. First, sets are labelled qualitatively (using linguistic variables such as "positive small", "negative large", "centre", "left", "right", and so on), and the elements of these sets are assigned varying degrees of membership. For instance, a temperature 22°C and a temperature 27°C may both be members of a set of "hot" temperature, although the temperature 27°C has a higher degree of membership (see Figure 4-5). Furthermore, any action or output resulting from a premise being true executes to a strength reflecting the degree to which that premise is true [VIO93].

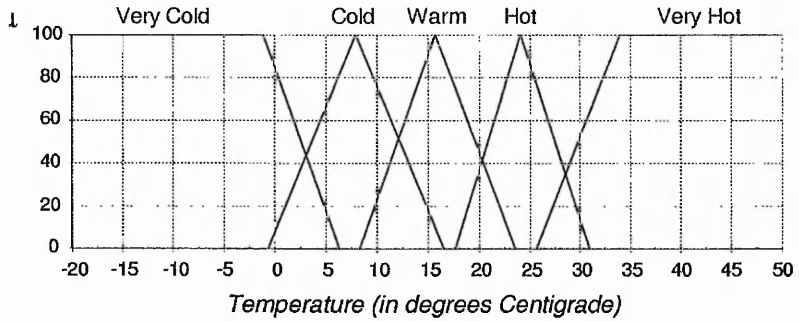


Figure 4-5: *Fuzzy sets for temperature*

Linguistic variables are defined as variables whose values are sentences in a natural or artificial language [ZAD73]. Thus, if high, pretty high, very high, etc. are values of density, then density is a linguistic variable. Natural language is a powerful tool allowing Man a comprehensive, but not very precise, description of reality. Its main power consists of an ability to employ vague notions. Every vague notion defines a certain class of objects whose boundaries are difficult to determine. Vague notions have so far been modelled by classical sets. It implies that borderline elements have to be put either into the set or outside it and appears to be the main reason for the often criticised inadequacy of mathematical methods in practice [NOV89].

Classical set theory is governed by logic that permits a proposition to possess one of two values: true or false. This logic does not accord well with the need to represent vague concepts. We see objects in shades of gray scale, not only in black and white.

The key idea in fuzzy set theory is that an element has a degree of membership in a fuzzy set. Consequently a proposition need not be simply true or false, but may be partly true to some degree. We assume that this degree is a real number in the interval [0, 100]. Consider the fuzzy set "tall". The elements are men, and their degrees of membership depend on their heights. For instance, a man who is 7 feet tall might have degree 100, and men of intermediate heights might have intermediate degrees. Different individuals will have

differing opinions whether a given man should be described as tall. A possible representation could be:

Height	Degree of membership
5'0"	0.0 (%)
5'4"	9.0 (%)
5'8"	33.5 (%)
6'0"	50.0 (%)
6'4"	83.5 (%)
6'8"	97.5 (%)
7'0"	100.0 (%)

According to this representation, the fuzzy set "tall" is defined by its domain. This is symbolised as

$$\text{tall: height} \rightarrow [0, 100]$$

"Height" is the domain or source of "tall", and [0, 100] is the target [NEG85]. It is not easy to use precise mathematical analysis to describe the behaviour of a system that is too complex or ill-defined, but it is using linguistic variables and fuzzy logic.

Information used in decision-making or reasoning processes can be uncertain, imprecise, or incomplete. In many cases of human reasoning, the reasoning ways based on vague statements and loose concepts are often considerable. Furthermore, some vague concepts are often represented by transformations of other vague concepts. Inference procedures that incorporate uncertainty are becoming more important in rule-based systems. In this project, the presented approach draws on this characteristic to cope with the flexibility problems described above. In the next section, the author will take a real case to explain how the researcher applied the fuzzy inference technique to detect the cutting paths within the distorted lace patterns.

4.3 Detection of First Cutting River

Two attempts have been made to find the first river within the lace pattern without prior knowledge of the pattern scanned. The first approach is based on *Pixel Intensity Directed Feature Extraction (PIDFE)* using traditional image processing methods. The second algorithm developed is based on *Fuzzy Pattern Recognition (FPR)* technique applying fuzzy reasoning rule-based method.

4.3.1 Pixel Intensity Directed Feature Extraction

In order to detect the river, a pixel intensity map is created to determine significant differences. This approach hinges on detection of large variations in intensity to highlight the river. Other contextual information such as pattern repeat cycle, river continuity and contour closing is used to speed up the process of feature extraction. This process can be broken down into the following tasks:

- Finding the edges of the lace.
- Converting the 256 level grey scale image into a bi-level bit map.
- Finding the repeating period of the pattern.
- Finding all probable rivers across the lace allowing for breaks.
- Finding the correct river from this list.

4.3.1.1 Edge finding

The black background surrounding the lace will be the darkest area of the image (the dark areas within the edges being lightened by the fine threads of the material). The first approach, therefore, may be to look for a sudden increase in point intensity. However, ambient lighting, shadows and reflections make this technique unreliable due to the difficulty in quantifying 'sudden' and 'increase'. Therefore, use can be made of the fact that these

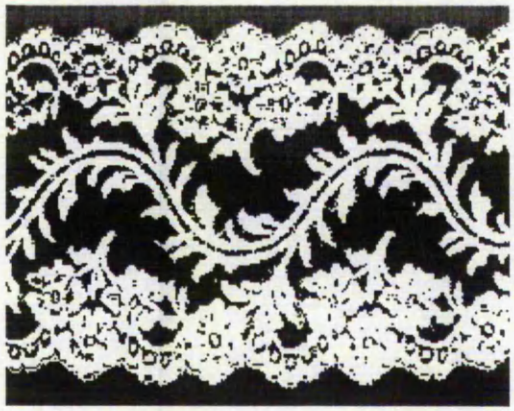


Figure 4-6: *Example of bi-leveled lace image*

phenomena tend to affect a large area and to work in terms of average intensities. By finding the average intensity of each vertical strip and examining which points lie above and below this threshold a clear pattern emerges with most of the area within the edges being brighter (certainly the thicker material actually on the edges) and all of the area outside the edges being darker. Thus, by finding the first and last points in a vertical strip which are above this value, the edge of the lace at that position is found. The small amounts of 'noise' in the image can be removed by looking for two consecutive points and the whole process can be speeded up by taking the average from a single strip from the middle and using this over the whole length.

A similar technique restricted by the boundaries found above is used to produce a bi-level representation of the lace pattern. The only difference being that the average must be calculated for each strip rather than using a single value (refer to Section 2.3.2). The points above and below the threshold are stored as 1 and 0 respectively in a bit map (Figure 4-6).

4.3.1.2 Finding the pattern repeat

The repeating period of the pattern (see Figure 4-7) must be found quickly and accurately from the above information. The technique used here is based on looking for

'landmarks' within the bit map, the most prominent being the large dark areas within the pattern. By defining large as being more than a set number of consecutive points within a strip (currently set to 16) and ordering the bit map in column order, the search can be speeded up by scanning the data for bytes containing zero and then scanning adjacent bits for the other required eight bits. This size ensures that whatever the alignment of the bits, the zero byte is always present.

Once these areas are found for each column, their widths and lengthwise offsets are calculated for rows eight bytes wide (aligned on the zero byte found earlier). By finding the distance between each dark area and all subsequent ones, the most frequently occurring distance will be due to the pattern repeating (if it were not for the cutoff at the edges of the image, this would occur for every dark area within the frame). This distance is the repeat period for the pattern and has been found to be accurate to within 2% of the actual value.

4.3.1.3 Finding river candidates

Taking a column down one side of the image it would be possible, excluding cross threads, to simply take each dark area and follow it to the right until:

- the other side is reached;

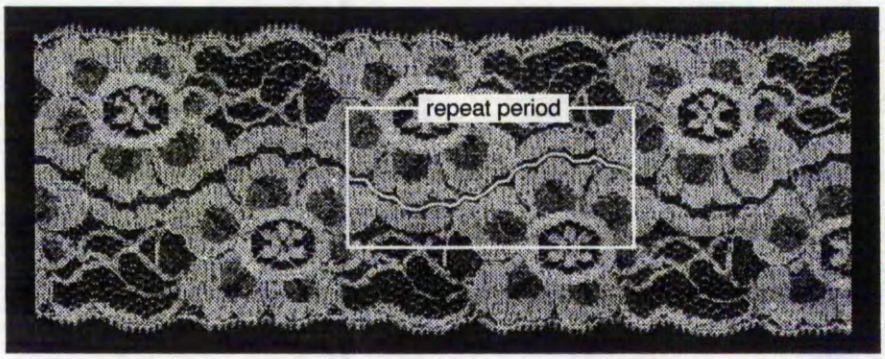


Figure 4-7: *The repeating period of the lace web*

- its boundary left the edge of the lace;
- it could proceed no further.

This would be marginally more complex if the river was allowed to loop back on itself but because of manufacturing techniques for the type of application considered here, this does not happen. As each river stretches along the length of the image, its top and bottom edge is stored. If this becomes too wide (more than twelve points for all samples tested) then the river can be discarded since this must be part of the pattern.

When the river can proceed no further by these means, it must initially be assumed that it has reached a cross thread. If this is not the initial search, then the previous river can be used to indicate the direction of the next point at the current stage of the pattern. However, for the first river the system must look for the nearest dark point without backtracking. A bias can be placed on this search depending on the direction of the previous points and distance from the middle. If the distance to this point is greater than a pre-determined threshold (an effective value has been found to be five points) then the river can be considered to have reached a premature end and is considered to be invalid.

4.3.1.4 Verification

The rivers found in the previous step can be tested against each other, as they are found, and only maintain the best case. Once a better one has been found, the previous best can be discarded and the new one takes its place. Two values are required to compare every two rivers. The first is a measure of symmetry within the width of the lace. This value is calculated by considering the coordinates of the points where the river changes direction (its outermost points) at the top and bottom of a cycle. These, when subtracted from their nearest edge and then subtracted from each other give a value for symmetry; the lower the value, the more symmetrical the river. The second is a measure of repeatability. In a perfect situation, the difference in river position one period further on from the start would be zero.

The smaller this difference, the higher would be the confidence. By comparing these two values, for each river, the best alternative can be found. When the best river has been found, the repeatability and symmetry values can be compared with a pre-determined threshold to ensure that it is satisfactory. If not, the machine must stop rather than cut a wrong path.

By using this approach, the detection of the river heavily depends on the feature of the repeated cutting path. The two extremes of the river should be equi-distant from their nearest edge, and after a distance equal to the repeat 'period' of the design. The river should be back at the same position relative to the two edges as it was when it started. As the lace pattern is distorted, these features of the river are no longer presented. This causes the dissatisfied results when the system applies this scheme to analyse the lace pattern contained the distortion bigger than 5-10 percent. In order to overcome the problems of material distortion due to the trimming and transporting processes, a *Fuzzy Reasoning Rule-Based Technique* is presented in the system.

4.3.2 Fuzzy Pattern Recognition

The scheme for applying fuzzy inference techniques to find the first river across the lace

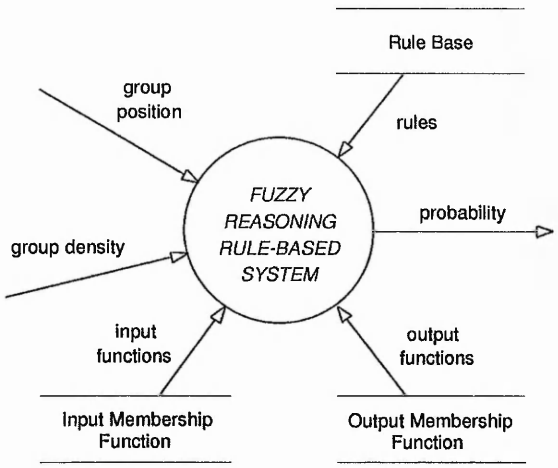


Figure 4-8: *Context diagram for system overview*

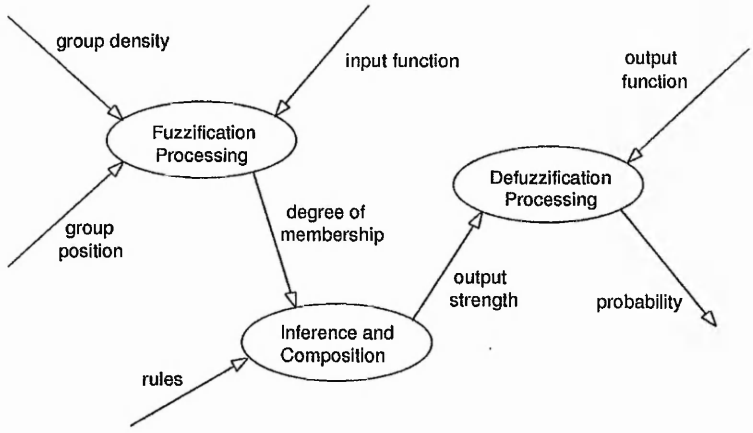


Figure 4-9: *Second level DFD for decision making process*

pattern with no previous knowledge can be broken down into the following tasks:

- Defining system input and output membership functions;
- Fuzzification process;
- Inference and composition;
- Defuzzification process;
- Verification.

Figure 4-8 illustrates the context data flow diagram of the system. This system reads two input variables (*Group Position* and *Density*) after each black pixel group has been processed. The fuzzification process then assigns a value to represent an input's degree of membership in one or more fuzzy sets. During inference and composition process, strengths are computed based on antecedent values and then assigned to the rules' fuzzy output. Finally, the defuzzification process employs compromising techniques to calculate the average weight for system output (Figure 4-9). These steps are described in detail as follows.

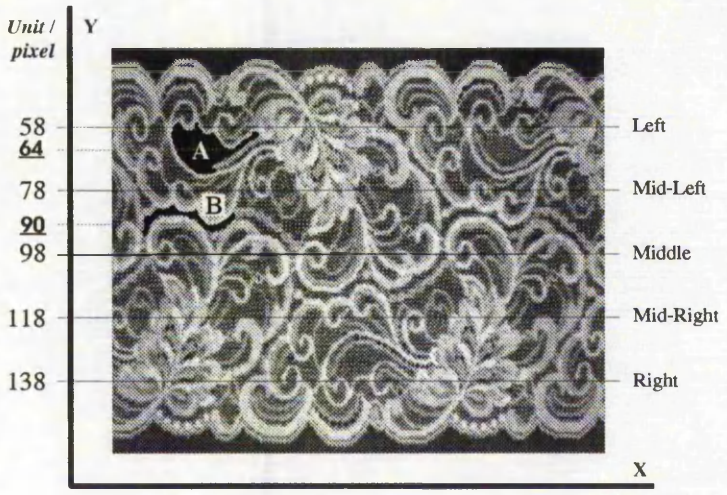


***Figure 4-10**: Corresponding positions for black pixel group A and B*

4.3.2.1 Defining system input and output membership functions

The degree of fuzzy membership is decided from overlapping sets of a membership function, which is defined normally based on intuition or experience. The pre-defined membership functions cover the entire range of values for system input and output, and will define a degree of truth for every point in the universe of discourse. As the system is tuned to accomplish desired responses to given inputs or output, it is accepted that membership functions change several times. Nevertheless, once the system is in operation, these

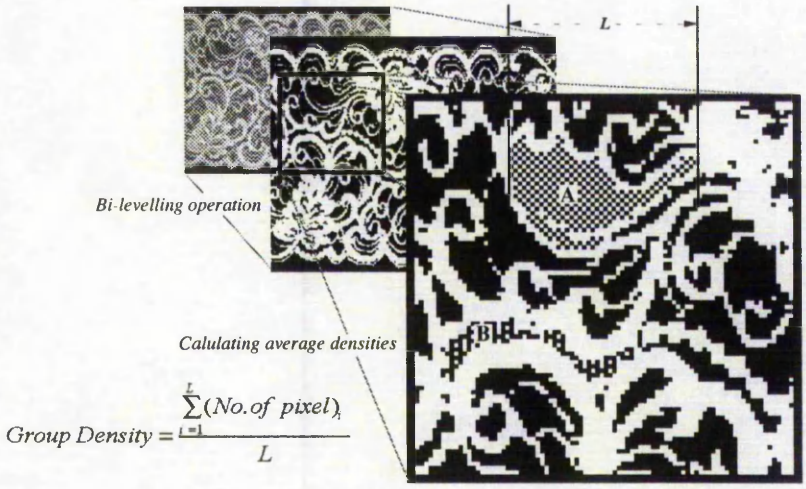


***Figure 4-11**: An example for calculating the group densities for group A and B*

membership functions will not be modified. The shapes and number of fuzzy-set membership functions we choose depend on parameters such as the required exactitude, steadiness and responsiveness of the system. Different shapes such as triangles and trapezoids are often employed to define fuzzy-set membership functions [ZAD88][ZIM87].

The objective here is to find the river along a lace pattern, by using linguistic variables to represent the common feature of the river shape in various lace patterns. These common features may be described as:

a) the *position* of the river is around the *centre* of the pattern;

b) the *density* of the river pixel group is not *high*.

From these linguistic descriptions, two system inputs, *group position* and *group density*, can be defined. By monitoring the position and density of the black pixel groups, as depicted in Figure 4-6, across a lace pattern, the fuzzy reasoning rule-based system can determine whether the pixel group is a probable segment of the river. Figure 4-10 and Figure 4-11 illustrate the two system inputs corresponding to an example lace pattern together with two candidate groups A and B.

Two initial experiments were carried out to define the system input and output

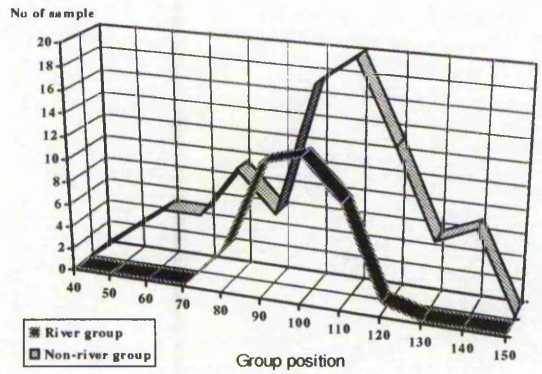


Figure 4-12: *Frequency histogram for the position of pixel groups*

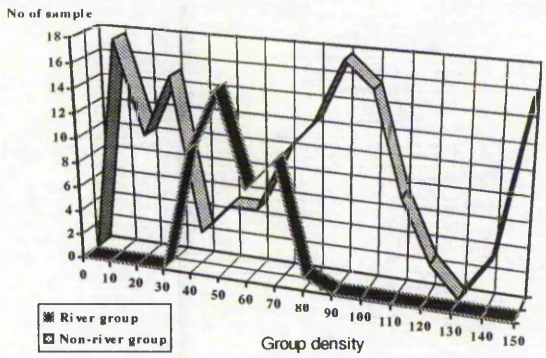


Figure 4.13: *Frequency histogram for the densities of pixel groups*

membership functions. Figure 4-12 and Figure 4-13 illustrate the frequency histograms which were taken from the experiments for defining input membership functions [ROB89][TUR84].

From these experimental results we can obtain a set of data from the *River group* part (see Figure 4-12 and Figure 4-13) to define the membership functions. Triangular membership function is most common and has proved to be a good compromise between effectiveness and efficiency. Overlapping between fuzzy-set boundaries is desirable and the

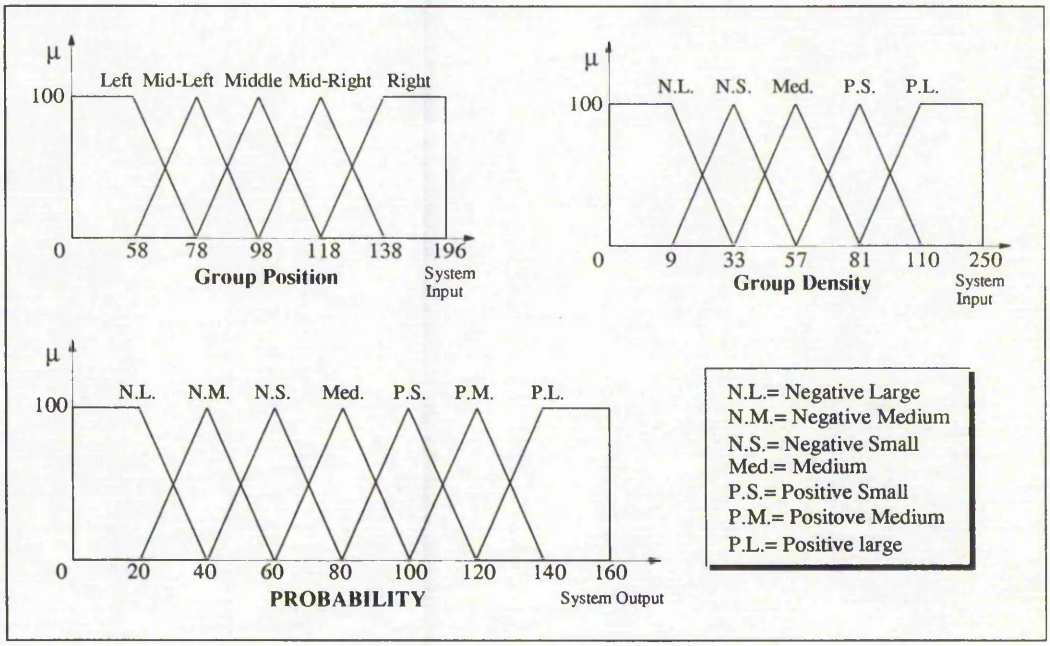


Figure 4-14: *System input and output membership functions*

key to smooth operation of the system. To simplify the procedure of defining fuzzy membership functions, an overlap of 50 percent between adjacent fuzzy sets is used in this experiment. In addition to each fuzzy set the central value and the slopes on either side are chosen. Figure 4-14 shows the fuzzy sets associated with the inputs and output of the system.

4.3.2.2 Fuzzification process

Fuzzification is the procedure of calculating an input value to represent a degree of membership in one or more fuzzy sets. This process uses two basic steps which are repeated for each system input. First, a crisp input has to be read and scaled to a value between 0 and 100. Second, the input must be translated to a degree of membership function. Figures 4-15 and 4-16 show two system inputs, *position* and *density*. Each value of system input has a degree of membership in each of these sets. Once the degrees of memberships are assigned, we can utilise these values to evaluate the rules.

4.3.2.3 Inference and composition

Fuzzified inputs are processed through a pre-defined set of rules using min-max evaluation to form fuzzified outputs. The author developed a set of rules that have the form

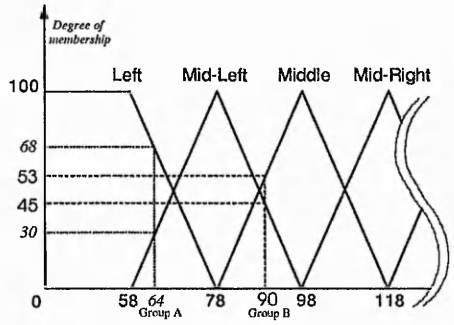


Fig. 4-15: *Fuzzy sets for "group position"*

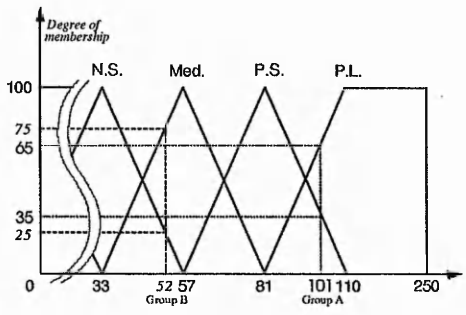


Fig. 4-16: *Fuzzy sets for "group density"*

Rule 1:	IF position is Left	AND	density is N.L.	THEN	probability is N.M.
Rule 2:	IF position is Left	AND	density is N.S.	THEN	probability is P.S.
Rule 3:	IF position is Left	AND	density is Med.	THEN	probability is P.S.
Rule 4:	IF position is Left	AND	density is P.S.	THEN	probability is N.M.
Rule 5:	IF position is Left	AND	density is P.L.	THEN	probability is N.L.
Rule 6:	IF position is Mid-Left	AND	density is N.L.	THEN	probability is P.S.
Rule 7:	IF position is Mid-Left	AND	density is N.S.	THEN	probability is P.M.
Rule 8:	IF position is Mid-Left	AND	density is Med.	THEN	probability is P.L.
Rule 9:	IF position is Mid-Left	AND	density is P.S.	THEN	probability is Med.
Rule 10:	IF position is Mid-Left	AND	density is P.L.	THEN	probability is N.L.
Rule 11:	IF position is Middle	AND	density is N.L.	THEN	probability is P.S.
Rule 12:	IF position is Middle	AND	density is N.S.	THEN	probability is P.M.
Rule 13:	IF position is Middle	AND	density is Med.	THEN	probability is P.L.
Rule 14:	IF position is Middle	AND	density is P.S.	THEN	probability is P.S.
Rule 15:	IF position is Middle	AND	density is P.L.	THEN	probability is N.L.
Rule 16:	IF position is Mid-Right	AND	density is N.L.	THEN	probability is P.S.
Rule 17:	IF position is Mid-Right	AND	density is N.S.	THEN	probability is P.M.
Rule 18:	IF position is Mid-Right	AND	density is Med.	THEN	probability is P.L.
Rule 19:	IF position is Mid-Right	AND	density is P.S.	THEN	probability is Med.
Rule 20:	IF position is Mid-Right	AND	density is P.L.	THEN	probability is N.L.
Rule 21:	IF position is Right	AND	density is N.L.	THEN	probability is N.M.
Rule 22:	IF position is Right	AND	density is N.S.	THEN	probability is P.S.
Rule 23:	IF position is Right	AND	density is Med.	THEN	probability is P.S.
Rule 24:	IF position is Right	AND	density is P.S.	THEN	probability is N.M.
Rule 25:	IF position is Right	AND	density is P.L.	THEN	probability is N.L.

Figure 4-17: *System rule base*

of

IF [*antecedent_one*] AND [*antecedent_two*] THEN [*consequence*]

which are listed in Figure 4-17. The antecedents of rules correspond directly to degrees of membership calculated during the fuzzification process. Each antecedent has a degree of truth assigned to it as a result of fuzzification.

In inference and composition processes, strengths are enumerated based on antecedent values and then assigned to the rules' output strengths. Figure 4-18 illustrates the actual fuzzy outputs calculated during the rule evaluation process for pixel group A. The strength of a rule is assigned the value of the weakest (*minimum*) antecedent. As more than one rule applies to the same specific action, the strongest (*maximum*) value of rules is used :

a) from Rule 4:

$$N.M._{rule\ strength} = min\ (antecedent\_one,\ antecedent\_two)$$

$$= min\ (68, 35) = \underline{35}$$

b) Rule 5:

$$N.L._{rule\ strength1} = \min (68, 65) = 65,$$

Rule1:	IF position is 68	AND	density is 0	THEN probability is N.M.
Rule2:	IF position is 68	AND	density is 0	THEN probability is P.S.
Rule3:	IF position is 68	AND	density is 0	THEN probability is P.S.
Rule4:	IF *position is 68*	AND	*density is 35*	THEN probability is *N.M.*
Rule5:	IF *position is 68*	AND	*density is 65*	THEN probability is *N.L.*
Rule6:	IF position is 30	AND	density is 0	THEN probability is P.S.
Rule7:	IF position is 30	AND	density is 0	THEN probability is P.M.
Rule8:	IF position is 30	AND	density is 0	THEN probability is P.L.
Rule9:	IF *position is 30*	AND	*density is 35*	THEN probability is *Med.*
Rule10:	IF *position is 30*	AND	*density is 65*	THEN probability is *N.L.*
Rule11:	IF position is 0	AND	density is 0	THEN probability is P.S.
Rule12:	IF position is 0	AND	density is 0	THEN probability is P.M.
Rule13:	IF position is 0	AND	density is 0	THEN probability is P.L.
Rule14:	IF position is 0	AND	density is 35	THEN probability is P.S.
Rule15:	IF position is 0	AND	density is 65	THEN probability is N.L.
Rule16:	IF position is 0	AND	density is 0	THEN probability is P.S.
Rule17:	IF position is 0	AND	density is 0	THEN probability is P.M.
Rule18:	IF position is 0	AND	density is 0	THEN probability is P.L.
Rule19:	IF position is 0	AND	density is 35	THEN probability is Med.
Rule20:	IF position is 0	AND	density is 65	THEN probability is N.L.
Rule21:	IF position is 0	AND	density is 0	THEN probability is N.M.
Rule22:	IF position is 0	AND	density is 0	THEN probability is P.S.
Rule23:	IF position is 0	AND	density is 0	THEN probability is P.S.
Rule24:	IF position is 0	AND	density is 35	THEN probability is N.M.
Rule25:	IF position is 0	AND	density is 65	THEN probability is N.L.

Figure 4-18: *Inference and composition for pixel group A*

from Rule 10 also

$$N.L.^{\text{rule strength2}} = \min(30, 65) = 30,$$

the maximum rule strength on fuzzy set N.L. is

$$N.L.^{rule\ strength} = max\ (65, 30) = \underline{65}$$

c) Rule 9:

$$Med.^{rule\ strength} = min\ (30, 35) = \underline{30}$$

In order to further improve the speed of this calculation, the Fuzzy Associative Memory (FAM) Bank [NED91] is applied to reduce the number of rules. Inspecting the FAM Bank (Figure 4-19), the following fuzzy system rule can be formulated:

from rule (A) indicated in Figure 4-19,

IF the *Group Position* is *Right*

AND the *Group Density* is *Positive Small*

THEN the *Probability** is *Negative Medium*

* *refer to glossary of terms*

Rule (A): IF Position is Right and Density is P.S.
THEN Probability is Negative Medium
Rule (B): IF Position is Near Mid. and Density is N.S.
THEN Probability is Positive Medium

Prob-ability		Position				
		Left	ML	Mid	MR	Right
Density	NL	NM	PS	PS	PS	NM
	NS	PS	PM (B)	PM	PM	PS
	Med	PS	PL	PL	PL	PS
	PS	NM	Med	PS	Med	NM (A)
	PL	NL	NL	NL	NL	NL

Figure 4-19: *Fuzzy Associative Memory (FAM) Bank to determine the probability*

This FAM Bank is comprised of 5 × 5 rules. We can reduce the 25 rules per FAM Bank to 11 rules per table by compounding the rules in the Bank. For instance, rule (b) indicated in Figure 4-19 merges three [*antecedent one*]s of the rules to take the form:

from rule (B),

IF the *Group Position* is *Near Middle*

AND the *Group Density* is *Negative Small*

THEN the *Probability* is *Positive Medium*

4.3.2.4 Defuzzification process

The defuzzification process is to convert its fuzzy outputs into a single raw or crisp output. There are more than 30 valid defuzzification methods. In these experiments, we choose the *"centre-of-gravity method"* which is a common and accurate defuzzification technique for resolving both the vagueness and conflict issues [ZAD88]. Figure 4-20 is used to illustrate the defuzzification of the output using the centre of gravity method:

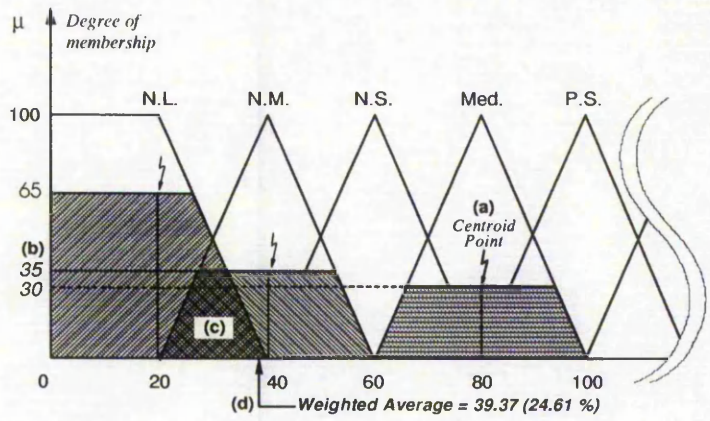


Figure 4-20: *Defuzzification process for pixel group A*

a) A centroid point on the x axis is found for each output membership function;

b) The membership functions are limited in height by the applied rule strength;

c) The areas of the membership functions are calculated;

d) The defuzzified outputs are derived by weighted averages of the centroid points and the enumerated areas:

$$Weighted\ average = \frac{\sum(shaded\ area \times centroid\ point)}{\sum(shaded\ area)}$$

By relying on the use of fuzzy inference technique, each black pixel group could be

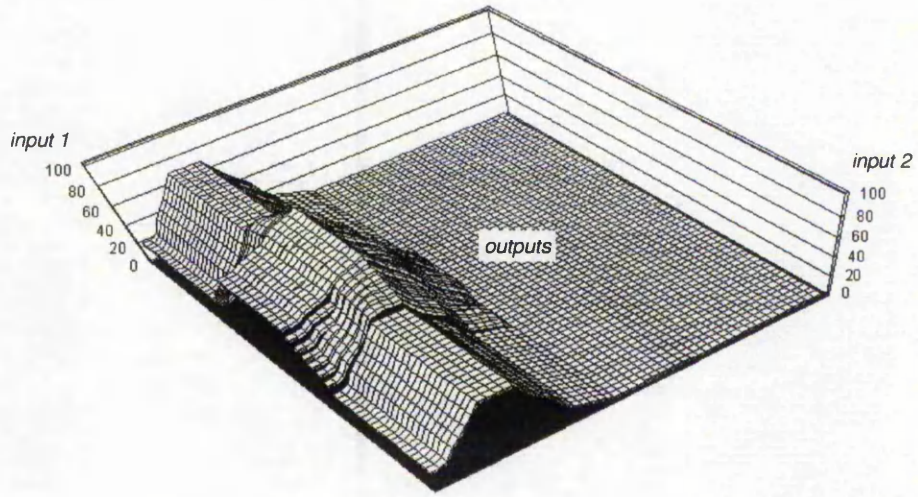


Figure 4-21: *Output pattern of the fuzzy engine*

calculated and assigned an average weight (*probability*). For instance, in Figure 4-11, the output value for group A is 39.37 (*24.61 %*) (refer to Figure 4-20), also the group B is 134.64 (*84.15 %*). Since the average weight of group A is only 24.61% (less than 50%), this means that the pixel group only has a 24 percent *probability* of being a segment of the river. It is, therefore, concluded that group A is not a part of the river. Figure 4-21 illustrates the output pattern of the fuzzy kernel using the FAM Bank (stated in Figure 4-19).

4.3.2.5 Verification

Once *all* the black pixel groups have been assigned a *probability value* (average weight), the pixel groups whose probability values are less than 80 (50%) are abandoned (see Figure 4-22). The verification process can then be broken down into the following tasks:

a) Calculate the distance between two adjacent groups;
b) If the distance is shorter than a specified value (set to six pixels long in these experiments) a network is built to record this path;
c) Continuously trace the distances between pixel groups while recording all the

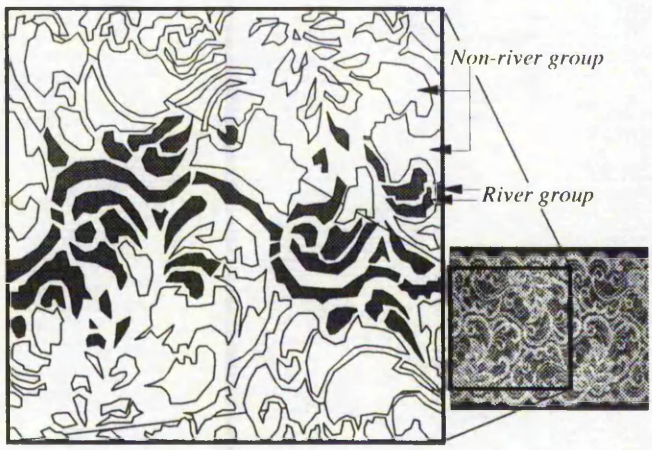


Figure 4-22: *Each probable river segments whose weights are bigger than 80 (50%)*

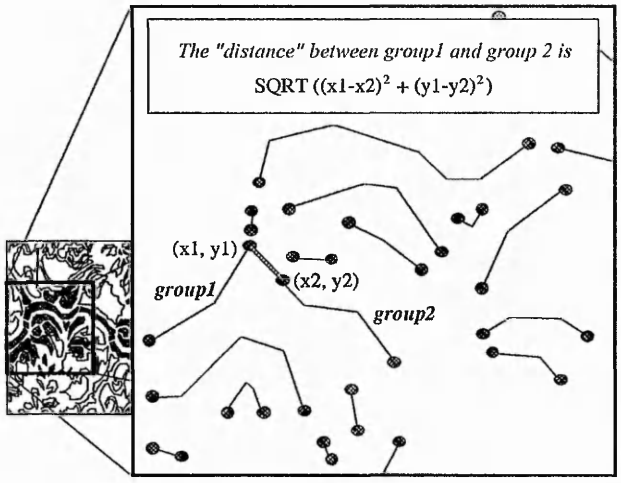


Figure 4-23: *Example for calculating the distance between pixel groups*

correct paths until a new pixel group reaches the border of the image (right hand edge of the frame);

d) Calculate the total probability values and divide by the number of the group in this path (*average probability*);

e) If the *average probability* is bigger than a specified value, (110 or 75% was used in the experiments) then the correct river has been found; if the average probability is less than this value, repeat step (c) to (e) until the correct river is located.

Figure 4-23 illustrates the computation of the distance between two adjacent pixel groups. By calculating the distances and tracing the average probabilities in all these segments, the river location, highlighted in Figure 4-24, can be pin-pointed.

4.4 Line Mapping Process

When the first cutting river in the lace strip is successfully detected, the extracted knowledge can be used to speed up the search in subsequent frames. In order to meet the real-time requirements of the system, instead of using traditional pattern matching

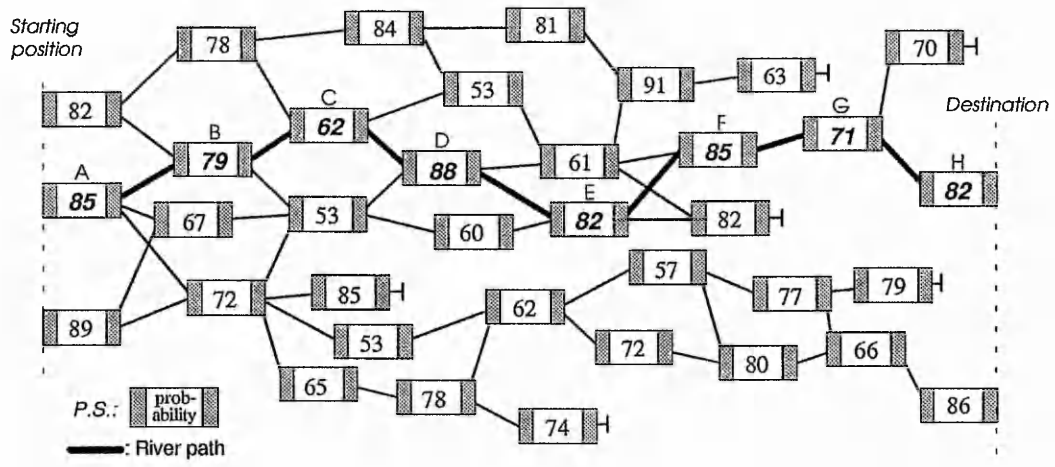


Figure 4-24: *Interconnection between each probable river segments*

techniques, a new approach called the *Line Mapping Method (LMM)* has been developed to achieve fast response and higher reliability. This approach is divided into the following processes:

4.4.1 Indicating and registering one repeat cutting cycle

A *centre line* is located by calculating the distance between the *upper* and *lower boundaries* shown in Figure 20. Three crossing points between the cutting river and the centre line are marked. The cutting path (river) between the intersections ① and ③ labelled

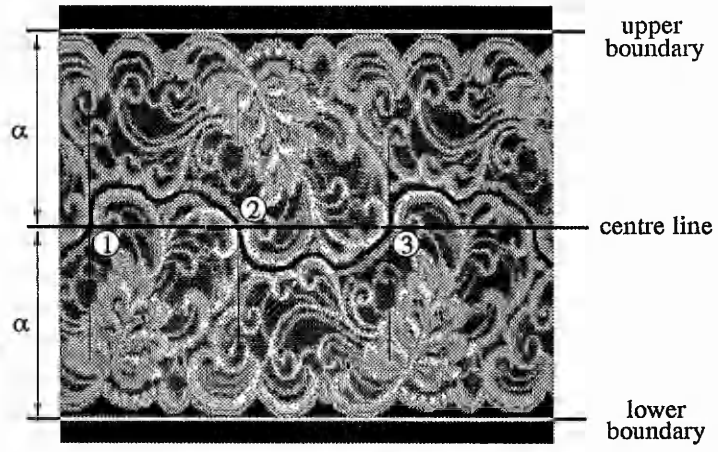


Figure 4-25: *Extracting a repeat cutting cycle*

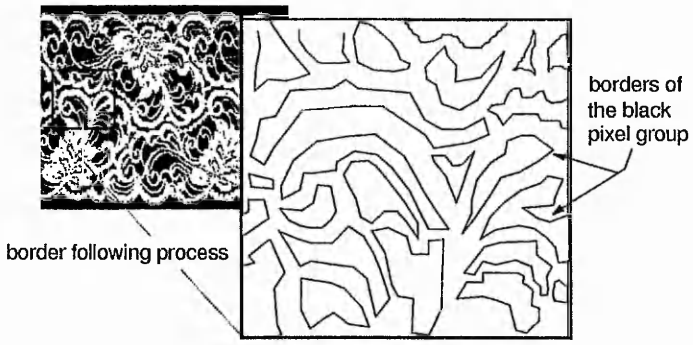


Figure 4-26: *Borders of the black pixel group*

in Figure 4-25 indicates a repeat cutting cycle, which acts as a *reference path* for detection of subsequent frames.

4.4.2 Capturing the following frame

The next frame of a 256 grey scale lace image is captured by the frame grabber from the CCD camera and temporarily stored in a memory block. An image thresholding operation is employed to transform the image into a black and white bitmap. This bi-leveled lace image is then applied for detecting the borders of the black pixel groups which may be candidates for river segments. As depicted in Figure 4-26, the border following technique is used to find the borders (outlines) of the potential river segments.

4.4.3 Mapping the reference path into the new frame

Since the lace strip is liable to distort as it is passed through the trimming mechanism, the *reference path* (river) is mapped onto the new frame for the detection of the next cutting river. With careful inspection it is clear, from Figure 4-27, that the two halves of the image do not completely match (the *reference path* is not completely within the river banks).

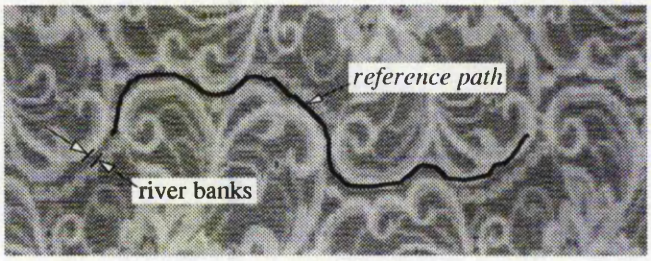


Figure 4-27: *Mapping the reference path into a new lace image*

A river, as stated previously, crosses from one side of the image to the other in a nearly unbroken sequence. Some allowance has to be made for cross threads produced as a result of the manufacturing process. These are *thick white threads* which cross the river at intervals and are indistinguishable from the material surrounding the river. For this reason, the detection must allow for small breaks in continuity.

The *LMM technique* has been developed for solving this problem. The detection will be started from the left hand side of the frame and ended at the right. As the matching point has

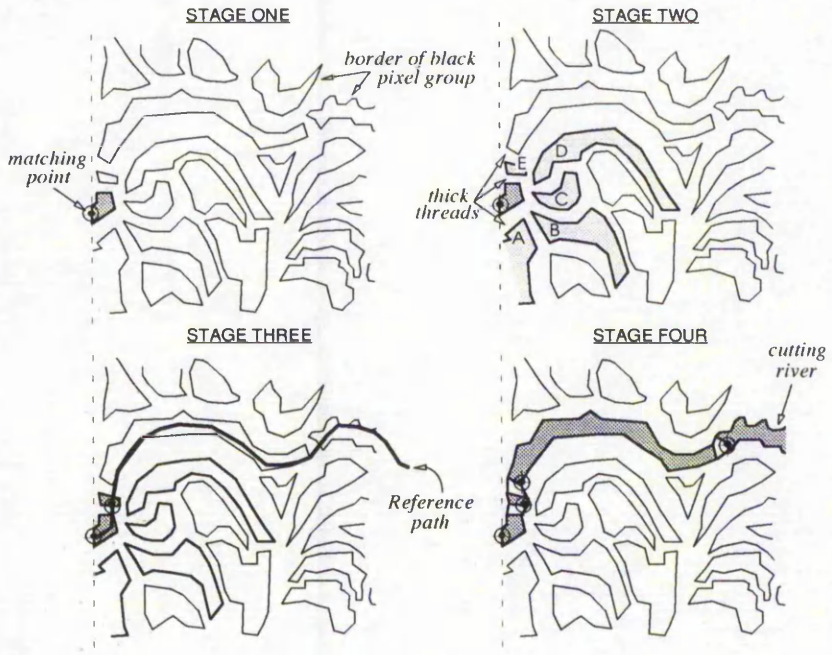


Figure 4-28: *Using the reference path for searching next cutting path*

been obtained (described in Section 4.5), the reference path is mapped onto the new frame to find the next border of the river. Several possible connecting borders can be found - A, B, C, D and E labelled in Figure 4-28 (stage two). The border closest to the mapped reference path is then chosen to become a part of the river (border E is selected in the example). Using the same method, the reference path is repeatedly employed to search the rest of the river segments until it reaches the end of the frame. After all the segments of the river have been found, lines between adjacent river borders are connected, as illustrated in Figure 4-28 (stage four), the entire river bank can be constructed. By using the detected river bank, a smooth line (the cutting path) can be created with in the centre of the river bank.

To summarise the scheme mentioned above:

1) Grab the first frame of the lace image;

2) Use the fuzzy reasoning technique to detect the first cutting river across the pattern;

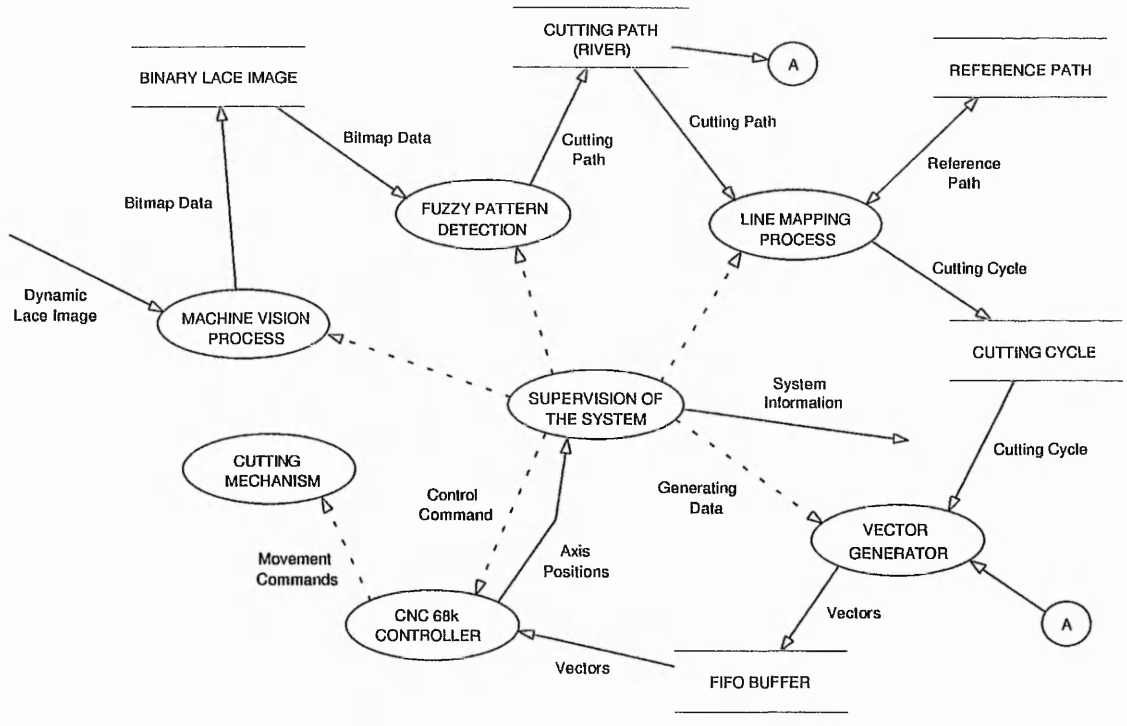


Figure 4-29: *DFD for the lace trimming system overview*

3) Find the intersections between the centre line and the cutting path;
4) Locate the capture and end of cutting points, respectively;
5) Generate machine movement data from the beginning to the end of cutting position;
6) Start trimming the lace pattern and continuously track the dynamic position of the cutter;
7) Capture the second lace image when the cutter reaches the capture point;
8) Use fuzzy technique to find the cutting river in the second frame for defining the shape of the *reference path*;
9) Download the machine movement data of the reference path;
10) Using the *LMM* method to map the *reference path* into the third and the subsequent frames of the lace image for fast detection of the repeat cutting path;
11) Continuously trim the strip of the lace into the desired pattern between frames.

Figure 4-29 illustrates the level 01 DFD for the system overview.

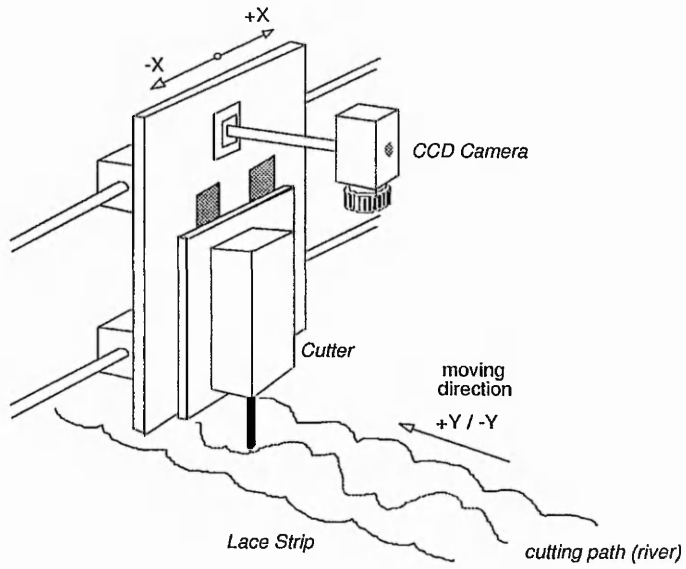


Figure 4-30: *The vision system and the cutter*

4.5 Supervision of the System

Since the CCD camera is mounted on both X and Y axes of the machine, the camera is moved with the cutter (Figure 4-30). The advantage of using such a construction is that the camera and the cutter are kept in a constant position relative to each other. Consequently, it is easy to calculate the real cutting position from the captured image, as well as to correct the errors between these two captured frames.

On the other hand, since the lace strip is transported past the vision system by the conveyor belt following the Y axis, the vision system has to consider the more complex two dimensional image shifting problem. Nevertheless, this *"look-and-move"* strategy yields more accurate results than the *"eye-to-hand co-ordination"* approach [WOL91], and also avoids small drift due to material length or missing steps of the motor(s). The strategy for analysing images moving in two directions and generating the vector data for machine control is discussed in the following sections.

4.5.1 Detecting the first river and finding the capture point for next frame

As the lace strip is transported past the field of vision, the first frame of the lace image is captured and temporarily stored in memory. After the fuzzy reasoning process, the cutting rivers across the lace pattern are found. The next stage of the system will then decide the *capture point* on the cutting path for the second lace image. When the machine is in operation, the camera is moving together with the cutter, so finding the position where the camera can capture a similar image for the *LMM* process is critical. As shown in Figure 4-31, a centre line can be drawn across the first frame, and an intersection between the cutting river can then be found. This position is engaged for grabbing the second frame of the lace image.

4.5.2 Generating machine movement data and grabbing the second frame

While the first cutting river has been detected, using the fuzzy reasoning method, the machine control data is generated and downloaded to the machine controller. The controller transforms the motion data into the real machine movement data and starts driving the cutter

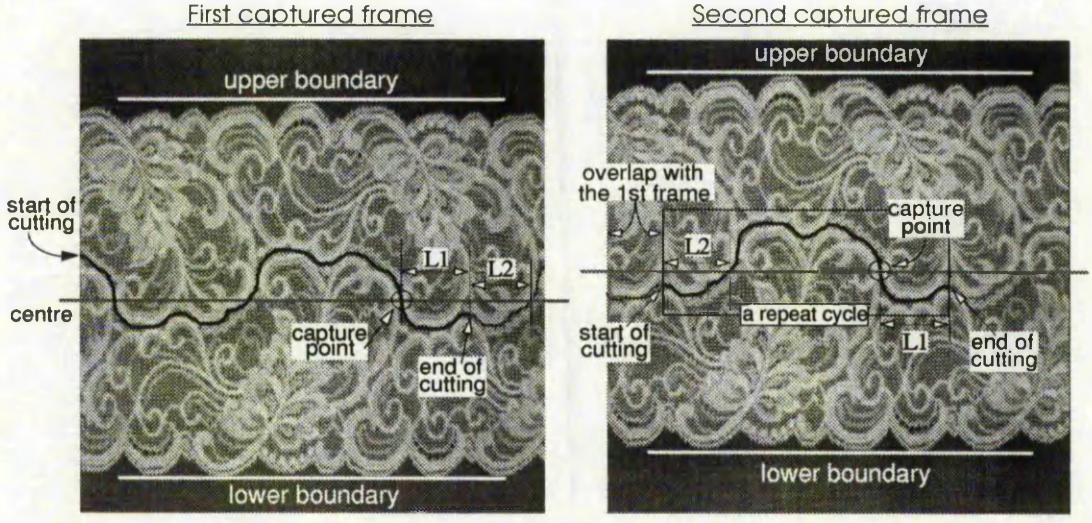


Figure 4-31: *Vision and cutting procedure*

to cut the strip of lace. When the machine starts cutting the lace strip, the controller simultaneously responds to the machine console with the current position on the XY axes. The machine console then continuously tracks the cutting positions until it reaches the *capture point* (shown in Figure 4-31, first captured frame). Consequently, the CCD camera is triggered to capture the second frame of the lace image which is stored into memory for processing.

Since the system takes approximately two hundred milli-seconds to find the next cutting river, this will stop the cutting process between two captured frames. To solve this problem, we simply add a quarter of the repeat cutting cycle (*L1*, between *capture* and *end of cutting points*, indicated in Figure 4-31) to the cutting path. Thus, while the machine is trimming past the *capture point*, the vision system grabs a frame as well as finding the cutting river before the machine actually ends trimming. This enables continuous operation of the system in real-time.

4.5.3 Finding the reference path and the next capture point

As the second lace image is stored in the memory, the fuzzy reasoning rule-based technique is, again, employed to find the second frame of the lace image. The second intersection with the cutting river can be designated as the *capture point* for the next frame. As the machine continuously trims the lace and reaches the *'end of cutting'* position in the first frame, the movement data of the second cutting path has already been produced and stored. Therefore, the machine could continuously cut the lace pattern through subsequent frames in an unbroken sequence.

L1 and L2 indicated in Figure 4-31 are taken from the first frame, and coupled to the second frame for determining the length and shape of the reference path (a repeat cutting cycle). After the reference path has been defined, its corresponding position with the centre

line (first pixel of this module) is then registered. This will be used for detection of subsequent frames.

4.5.4 Line mapping operation

After the *reference path* has been determined, the extracted knowledge can be used to speed up the search for the river in subsequent frames. Figure 4-32 shows an example of mapping the reference path into the following frame. Utilising the *Line Mapping Method (LMM)*, the new cutting river across the lace image can be successfully and quickly detected.

4.6 Experimental Results

Various experiments were carried out to investigate the effectiveness of this method. Numerous lace patterns were employed for detecting the river location. All cutting paths across the patterns were successfully found. The time taken to isolate the river and produce cutting path depends on complexity of the pattern. Time taken for most kinds of motif, using the fuzzy reasoning rule-based technique, is typically about 300 milli-seconds using an Intel 80486 processor running at 66 MHz. Nevertheless, in the case of a very few intricate lace

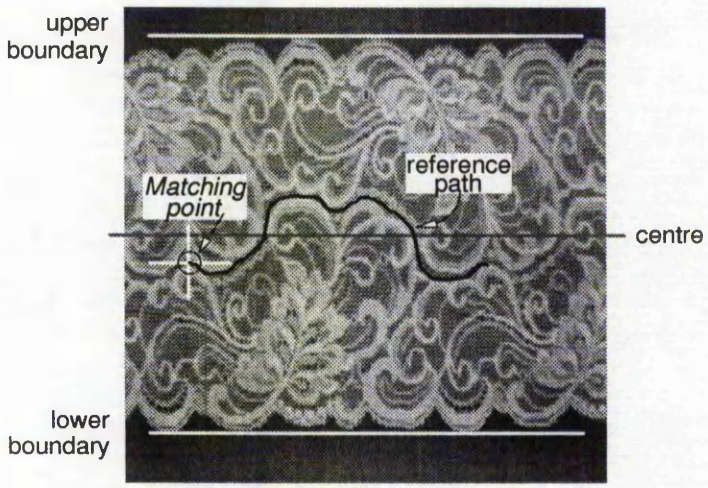


Figure 4-32: *Mapping the reference path into subsequent frame*

Figure 4-33: *Example A of river extraction*

patterns (e.g. Figure 4-33), up to 1.5 seconds is required.

Once the river path on the first frame is found, this knowledge can be utilised by the LMM to speed up the detection for the river in subsequent frames in real time. The time to detect a repeat cutting path using LMM is dependent on how complex the motif is, the length of one repeat cutting cycle and the distortion of the pattern. On most kind of lace patterns detection time is about 150 to 200 milli-seconds. The frame grabber digitises a incoming video signal at a rate of 30 frames per second. Typically a repeat cutting cycle of the lace strip is around 9 to 15 cm. Therefore the speed for tracking the lace pattern using the LMM is approximately 25 to 35 meters / minute (higher performance could be obtained with a faster computer). Some sample lace patterns together with the resulting river path are

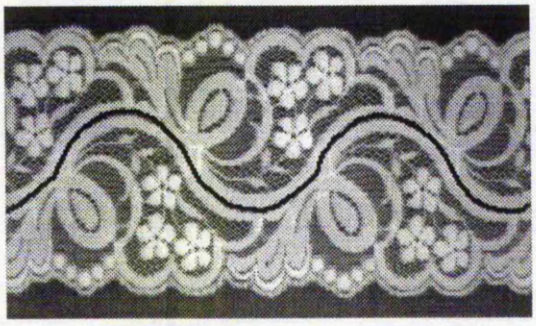


Figure 4-34: *Example B of river extraction*

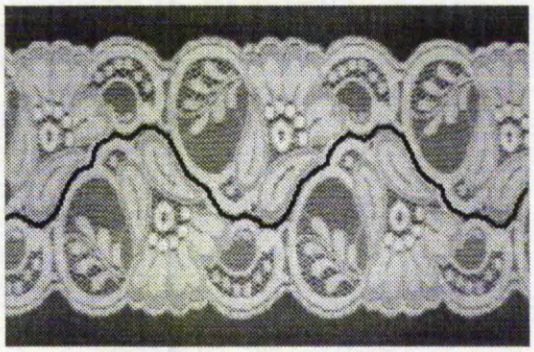


Figure 4-35: *Example C of river extraction*

shown in Figures 4-33 to 4-35.

The strip of lace is likely to stretch or contract while it is passed through the machine via the feed mechanism. Several experiments have been carried out for investigating the capability of this approach. Various kinds of lace patterns have been examined under the following status: 1) Non-distortion, 2) Contraction, 3) Stretch, and 4) Un-parallelism.

1) No-distortion

Under this condition, all cutting paths across the lace patterns were successfully found.

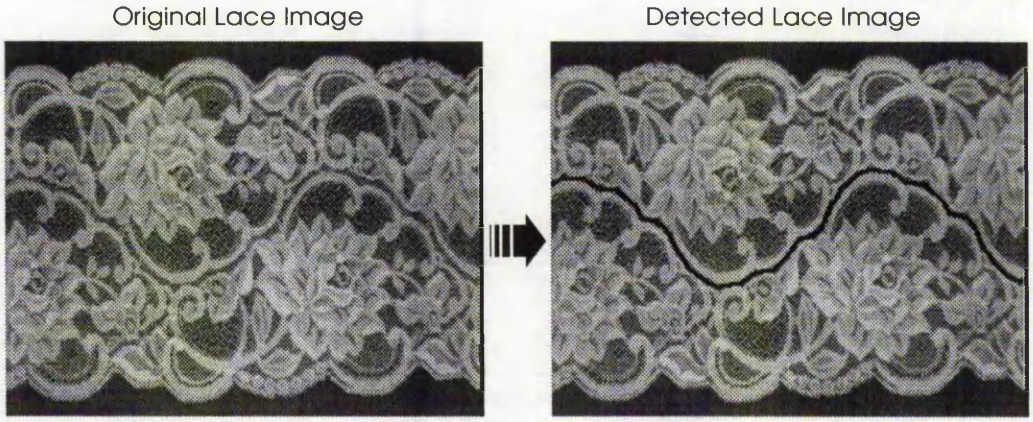


Figure 4-36: *Un-distorted lace pattern*

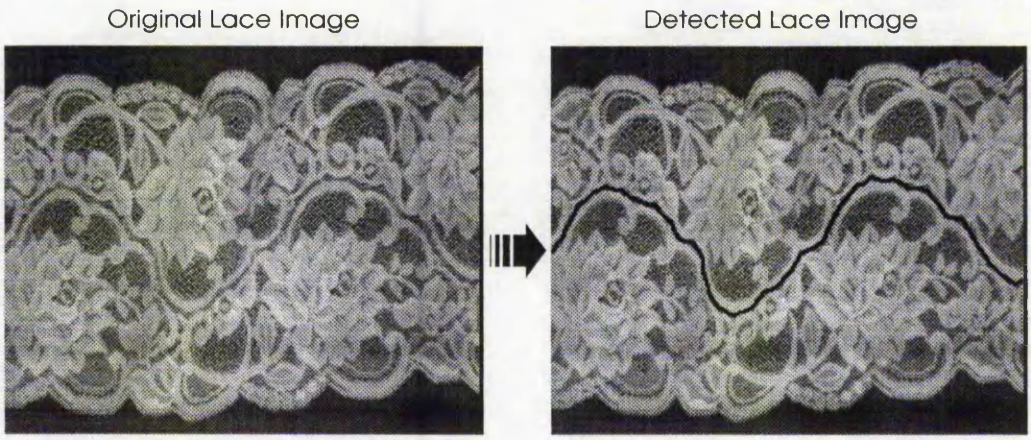


Figure 4-37: *Lace pattern under 30% contraction (successfully detected)*

Figure 4-36 illustrates a typical lace pattern as well as its detected cutting path.

2) Contraction

Figure 4-37 and Figure 4-38 show that the lace motifs have been contracted with 30 per-cent and 40 per-cent of the pattern. The cutting paths within these two frames have been successfully detected by applying the novel techniques. On the other hand, when we use traditional image processing methods [SHE94a] to examine these distorted patterns, according to the experimental results, none of them could find the river, or even pinpoint the

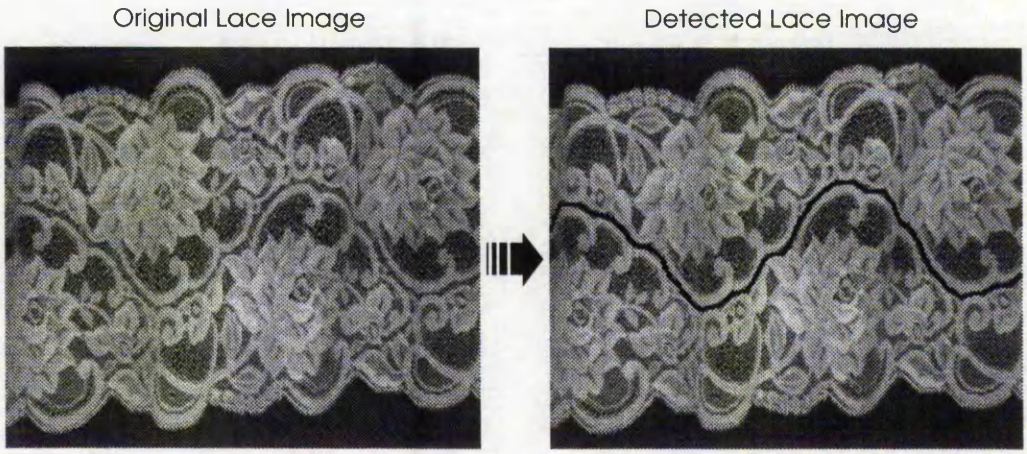


Figure 4-38: *Lace pattern under 40% contraction (successfully detected)*

wrong paths.

The feature of the lace motif under 50 per-cent of contraction is illustrated in Figure 4-39. With careful inspection of this picture we can find that parts of the river banks are overlapped, it is difficult to find a nearly unbroken river across the strip of the lace. In the real working situation, although the skilled human operator could find the cutting path from the contracted lace pattern, it would be impossible to separate this pattern correctly by using knife or lace cutter. Consequently we can ignore this situation in our experiments.

3) Stretch

Lace was stretched lengthwise in order to emulate stretch resulting from lace transport. Figure 4-40 shows that the lace pattern has been stretched as much as possible, and its detected river across the pattern. Typically, about 15 to 30 per-cent of lace can be stretched, depending on material and patterns.

Compared with the previously reported method [SHE94a], the fuzzy logic based

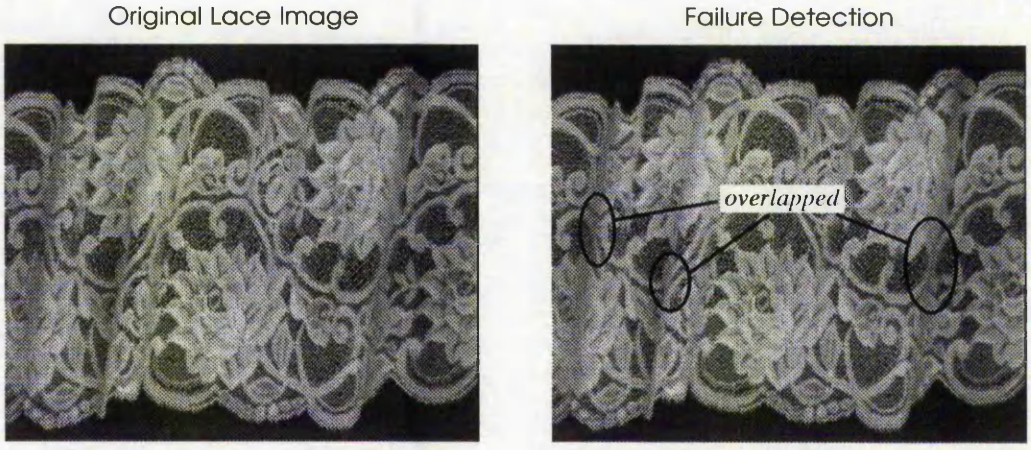


Figure 4-39: *Lace pattern under more than 50% contraction*

(fail to detect the correct cutting path)

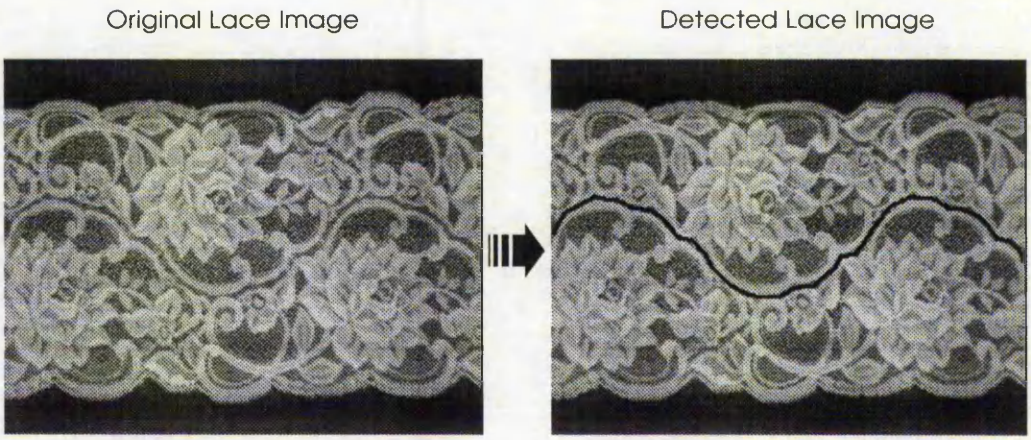


Figure 4-40: *Lace pattern under maximum stretch*

approach (FPR: Fuzzy Pattern Recognition) is more effective. The traditional image processing technique (PIDFE: Pixel Intensity Directed Feature Extraction) for finding the river heavily depends on the repeat of the lace pattern. In other words, the two extremes of the river should be equi-distant from their nearest edge, and after a distance equal to the repeat period of the design, the river should be back at the same position relative to the two edges as it was at the start. When lace is distorted, these features of the river are absent. That is why the conventional method fails when strips of lace are slightly distorted (5- 10%).

4) Un-parallelism

As the lace strip is placed on the cutting bed by an operator, a small degrees of deviation in the angle between the lace and the conveyor belt will normally occur. Figure 4-41 depicts the skewed lace patterns captured by the camera. Applying the methods developed, the cutting paths in the lace patterns up to ± 20 degrees of deviation in the angle can be successfully detected (see Figure 4-41). According to the experiences of the researcher, as the angle of the deviation between a lace strip and the conveyor belt is larger than 20 degrees, the lace strip is not possible to be rolled on successfully by the conveyor system. Accordingly we only consider that the deviation angle of the lace strip is not larger than ± 20 degrees in our experiments.

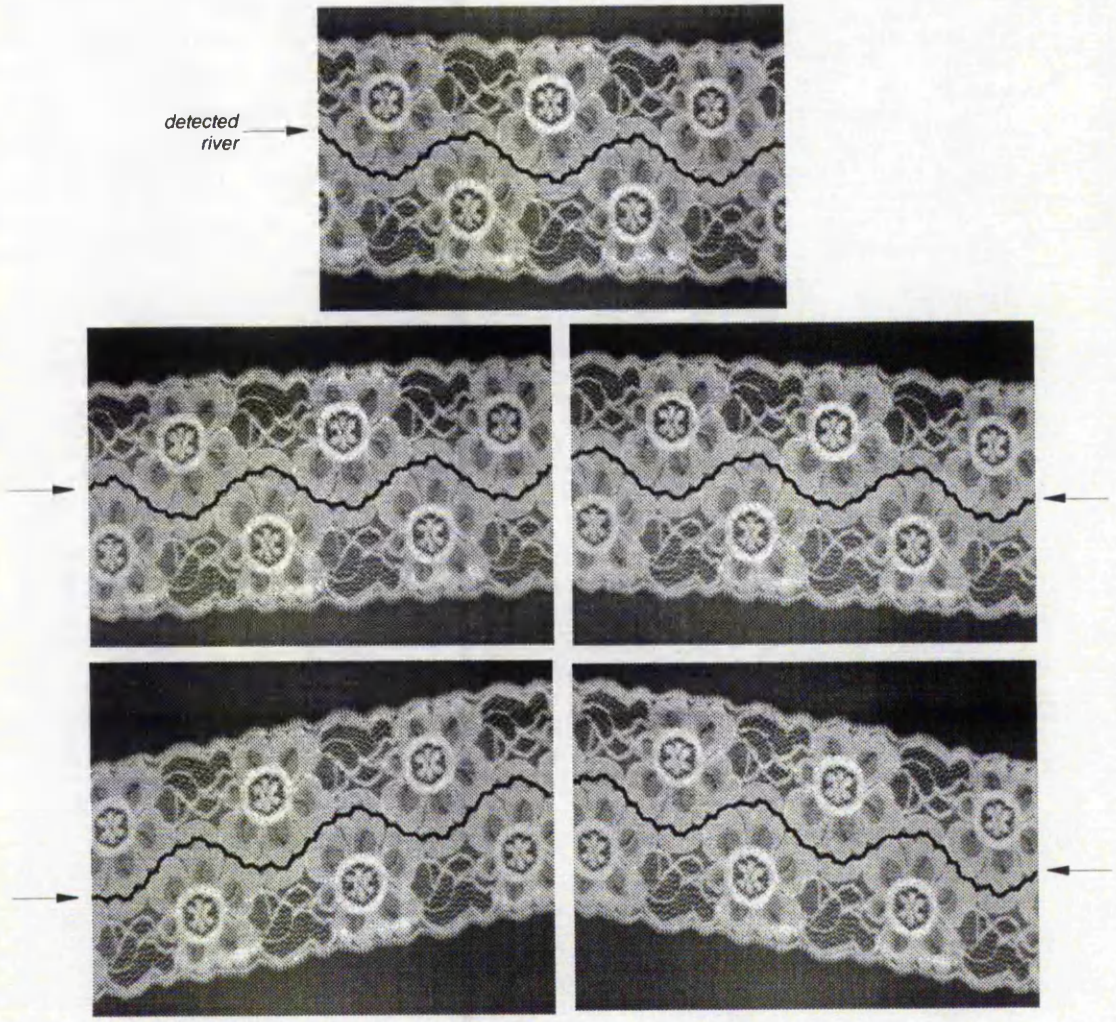


Figure 4-41: *Detections of cutting paths within the lace patterns up to ± 20 degrees of deviation*

Experimental results indicate that the objectives have mostly been fulfilled. The system requires no prior knowledge of any particular lace pattern or any training. A combination of FPR technique and the LMM can be used to successfully detect the cutting river within various lace patterns in real time. Compared with the conventional image processing methods [KIN93][RUS88][SHE94], it is not only easier to design and implement the system, but also more effective in coping with distortion. Furthermore it does not require any training or prior knowledge of the lace pattern.

4.7 Summary

This chapter describes the techniques developed to detect the cutting paths within various lace patterns in real time. The author has described two attempts based on *Pixel Intensity Directed Feature Extraction (PIDFE)* using traditional image processing methods and *Fuzzy Pattern Recognition (FPR)* technique applying fuzzy reasoning rule-based method to detect the first river within a lace pattern without prior knowledge of the pattern scanned. According to numerous experimental results, the FPR technique shows to be more effective in coping with distortion than the PIDFE method.

As the first river has been successfully found, the extracted knowledge can be used by the *Line Mapping Method* to quickly detect the cutting paths within the subsequent frames. By relying on this method, it can achieve fast response and higher reliability. The combination of the Fuzzy Pattern Recognition Technique and the Line Mapping Method can be used to successfully detect the cutting river within various lace patterns in real time.

The techniques developed to supervise the vision system and the cutting process have been described. The CCD camera has been mounted on the X axis of the cutting mechanism in which the "look-and-move" strategy yields more accurate results than the "eye-to-hand co-ordination" approach, and also avoids small drift due to material length or missing steps of the motors.

5. TIGHTLY COUPLED VISION AND CONTROL SYSTEM

Chapter 5
TIGHTLY COUPLED VISION AND CONTROL SYSTEM

5.1 Introduction

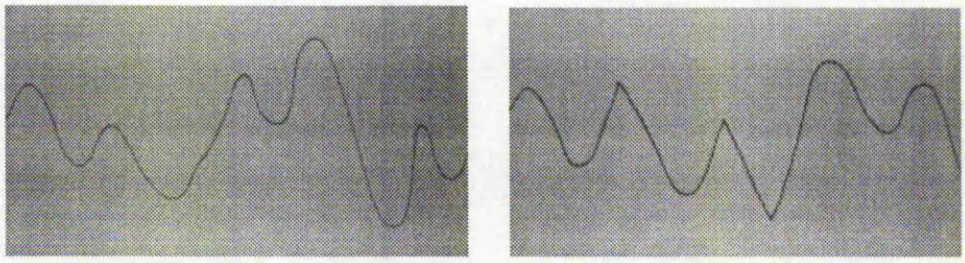
Lace is most commonly manufactured by a knitting process in a web up to 3.5m wide with multiple pattern repeats across its width [KIN93]. The web is separated into lengths, which involve 'scalloping' - cutting along the edges of the pattern. This is performed by operators who guide the lace onto rotary knife cutters. Lace is deformable (it may even be manufactured with elastic yarns to enhance its elasticity) and the variability of the manufacturing process produces large tolerances on the pattern dimensions (up to ± 10%, sometimes in combination with pattern skew). Further practical problems can result from distortion of the lace as knitting tensions are released during the separation process.

For economic reasons a lace web speed of 1m/s is desirable which requires very rapid processing of image data to provide real-time control of the cutting mechanism. The task is made difficult by the deformable nature of the lace whose dimensions vary with tension and

manufacturing conditions and the changes in the pattern caused by the release of tension in the lace structure as it is cut. The extent of stretch varies depending on material and the structural pattern of lace. Specially when *tactile cutters* are used, cutting forces may result in changing the image. This contributes to the complexity of real-time pattern recognition problem.

Work has been reported using a guided laser cutter to separate the deformable materials [KIN93]. This system uses multiple digital signal processors to acquire and process image data from a line-scan CCD camera. Control information is then generated to allow cutting of the web along the tracked path (based on pre-defined template poses) using a CO_2 laser beam deflected by a galvanometer mounted mirror. Although the use of laser reduces this deformation, distortion due to mechanical feed misalignments persists. Changes in the lace pattern are caused also by the release of tension in the lace structure as it is cut. In order to tackle the problem of distortion due to material flexibility in general, a novel approach using *inexact algorithms*, i.e., *fuzzy logic*, *neural networks* and *neural fuzzy technique*, have been developed and described here. We proposed a tightly coupled vision control system using a *tactile cutter* (for cost effectiveness). By employing the pre- and post-processing vision systems with the intelligent machine console, it is possible to monitor the process and generate on-line information for the controller and hence overcome the flexibility problem.

In this programme of work it is proposed to use a strip of paper on which has been



Example A Example B

Figure 5-1: *The curve patterns on a strip of paper captured from a CCD camera*

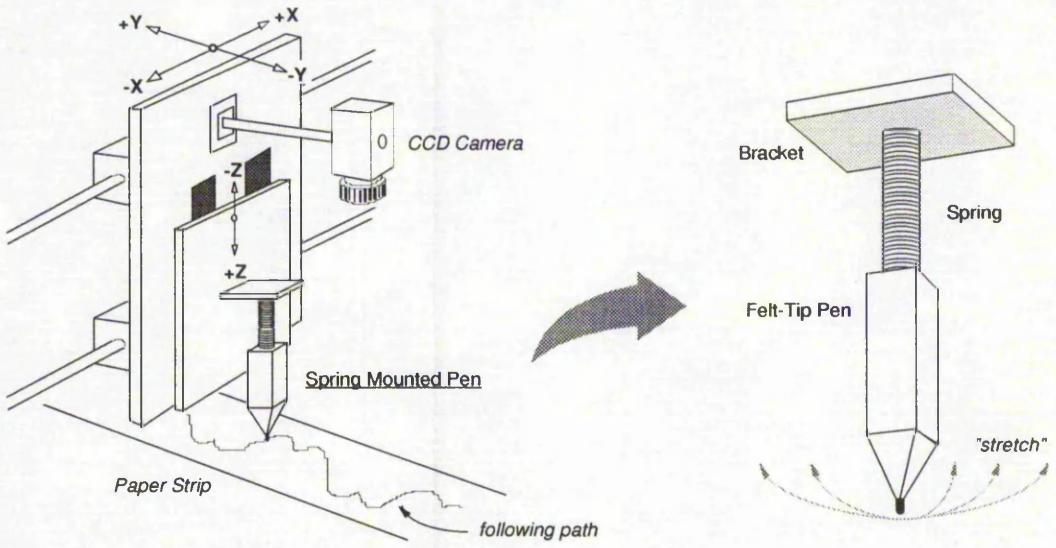


Figure 5-2: *Spring Mounted Pen (SMP) connected with the testing rig*

drawn a curve to simulate the river extracted from the lace patterns (Figure 5-1). A *spring mounted pen (SMP)*, which is a pen with a spring on the top connected with the Z axis of the cutting / drawing mechanism, is employed to emulate the movement (distortion) of the lace strip due to the cutting forces caused by a tactile cutter. Figure 5-2 shows the schematic of the SMP together with the axes of the testing rig.

The main emphasis of the work is to investigate the use of inexact algorithms in order to overcome the problems of lace distortion. Furthermore, it is proposed to use the inexact methods as the basis for achieving optimal quality which satisfies visual demands rather than engineering precision. Employing inexact algorithms, potentially allows development of methods for detecting unsatisfactory or faulty operation which may be applicable to similar manufacturing processes such as knitting and garment producing, etc.

In the following sections, first, we are going to discuss the concepts of *Neural Networks* and *Neural Fuzzy* theories, respectively, and then explain in detail how they have been implemented by means of using C programming language in an IBM compatible personal computer.

5.2 Neural Networks

Artificial neural networks or simply "neural nets" go by many names such as connectionism, parallel distributed processing, connection science, neural computing and neuromorphic systems. Whatever the name, all these models attempt to achieve good performance via dense interconnection of simple computational elements. In this respect, artificial neural net structure is based on our present understanding of biological nervous systems. A simple, three-layer back-propagation model comprising two input neurodes (*active units in a neural network*)[EBE90], five hidden neurodes and three output neurodes is illustrated in detail in Figure 5-3. Designing artificial neural networks to solve problems and studying real biological nets may also change the way we think about problems and lead to new insights and algorithmic improvements.

5.2.1 Introduction

Artificial neural net models have been studies for many years in the hope of achieving human-like performance. These models are composed of many nonlinear computational elements operating in parallel and arranged in patterns reminiscent of biological neural nets. Computational elements or nodes are connected via weights that are typically adapted during

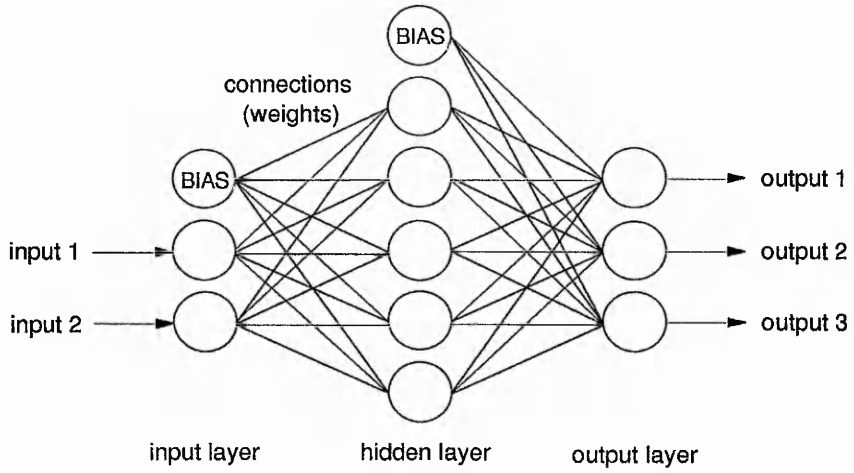


Figure 5-3: *Back-propagation network structure*

use to improve performance. Aleksander and Morton [ALK90] have the following definition:

> *Neural computing is the study of networks of adaptable nodes which, through a process of learning from task examples, store experimental knowledge and make it available for use.*

A neural network is a processor of information consisting of simple processing elements connected together. Each processing element is a vary simple model of a neuron in the brain: hence the term neural networks. Thus a neural network could be described as mankind's attempt to create an artificial brain. Present neural networks try to mimic some processes in the brain in order to harness its ability to infer and induce from incomplete or confusing information.

What makes these networks powerful is their potential for performance improvement over time as they acquire more knowledge about a problem, and their ability to handle fuzzy real world data. That is, a network 'taught' certain data patterns, is able to recognise both these patterns and those which are similar: the network is able to generalise.

Work on artificial neural networks has a long history. Development of detailed mathematical models began more than 50 years ago with McCulloch and Pitts, Hebb, Minsky, Rosenblatt, Widrow until 1969. Unfortunately, the claims by Rosenblatt as to what his machine could do were somewhat exaggerated, and in 1969 Minsky and Papert wrote a book, *Perceptrons* [MIN69], which showed that there were various problems with these neural networks and so the subject went out of fashion until the mid 1980s. During this time, however, various people were still working in the field, including Aleksander, Kohonen, Hopfield, and the PDP group including Rumelhart, McClelland and Hinton. Neural networks have become fashionable again because it has been realised that there are limitations as to what Expert Systems can do, and the success of Hopfield and the PDP

group. In 1986 Rumelhart and McClelland published *Parallel Distributed Processing* [RUM86] in which it is shown how the objections of Minsky and Papert could be overcome, and work in the field has flourished.

5.2.2 Principles

The potential benefits of neural networks extend beyond the high computation rates provided by massive parallelism. Neural nets typically provide a greater degree of robustness or fault tolerance than von Neumann sequential computers because there are more processing nodes, each with primarily local connections. Damage to a few nodes or links thus need not impair overall performance significantly. Most neural net algorithms also adapt connection weights in time to improve performance based on current results. Adaptation also provides a degree of robustness by compensating for minor variabilities in characteristics of processing elements. Traditional statistical techniques are not adaptive but typically process all training data simultaneously before being used with new data. Neural net classifiers are also non-parametric and make weaker assumptions concerning the shapes of underlying distributions than traditional statistical classifiers. They may thus prove to be more robust when distributions are generated by nonlinear processes [LIP87].

A neural network is not programmed to complete a given task, rather it adapts and acquires knowledge (learning process, which can be supervised or unsupervised) over time in order to complete the task. Instead of performing a program of instructions sequentially as in a von Neumann computer, neural networks explore many competing hypotheses simultaneously using massively parallel nets composed of many computational elements connected by links with variable weights. Once the network has learned to perform a task, it can then be set to undertake that task. For instance, a network which is used to classify a massive set of data would first be taught what those sets are categorised by which class, and then would be put in the mode whereby it reported whether the set it was being shown was one it had been taught.

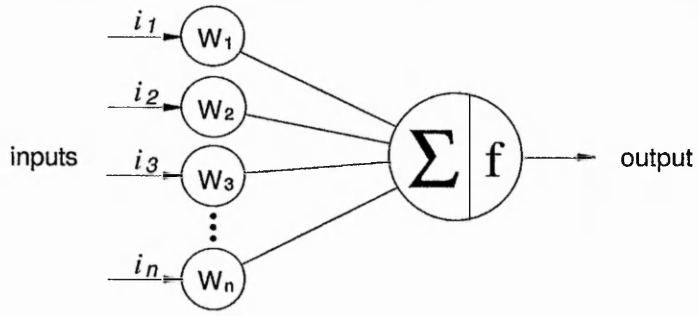


Figure 5-4: *Basic model of Neuron*

A neural network consists of simple processing elements connected together. There are various ways in which these elements can be connected, in single or multiple layers, fully or partially connected, etc. Besides there are very many forms of neural network of which the main types are the 'Perceptron' types, Hopfield nets, Boltzmann machines, Weightless or n-tuple nets, Kohonen nets, the neocognitron and ART classifiers. 'Perceptron' networks is one of which are used most, and are more applicable in the area covered by the investigations reported here, and it is described next.

5.2.2.1 Perceptron networks

In these networks the main element is an extension of the McCulloch and Pitts neuron; this is shown in Figure 5-4. The output of the element is some function of the weighted sum of all the inputs. Sometimes the output value is used directly, or the output is processed by, say, a threshold or the sigmoid function which will be explained later. In some networks it is important to know that the element has 'fired' (in the same way that a neural in the brain produces an active output), and this occurs if the weighted sum exceeds a given threshold. For other networks, the actual value of the output is required.

Neural networks are specified by the net topology, node characteristics, and training or learning rules. These rules specify an initial set of weights and indicate how weights should

be adapted during use to improve performance. Learning is the process of deciding what the values of the weights should be. This is often achieved by comparing the output of the element with what it should be (this is a form of supervised learning) and adjusting the weights appropriately. Normally the *Widrow-Hoff Delta rule* is used to adjust weights.

A single layer network of such neurons can perform simple functions. However, one of the criticisms raised by Minsky and Papert was that such a network could not solve a non linearly separable problem: that is one where one straight line cannot separate opposing classes. However, a multiple layer network, which is shown in Figure 5-3, can solve such problems.

In learning mode, the network is separately presented with the data in the training set, both the required input and output for each item in the set, and the weights adjusted accordingly using the generalised delta rule, by propagating errors between the actual output and the desired output back through the network. This can take time as the whole data set must be presented many hundred times. This is a severe disadvantage of the method. However, a great many workers in the field use back propagation learning on multi layer perceptrons.

In this project, the author has employed the most widely used of the neural network paradigms - back propagation (or called back-error propagation) networks which has been applied successfully in applications studies in a broad range of areas.

5.2.2.2 Back-error propagation

Back-propagation is one of the easiest networks to understand. Its learning and update procedure is intuitively appealing because it is based on a relatively simple concept: If the network gives the wrong answer, then the weights are corrected so that the error is lessened and as a result future responses of the network are more likely to be correct.

Back-propagation is a tremendous step forward compared to its predecessor, the perceptron. The perceptron was limited to only two layers of processing units, with only a single layer of adaptable weights. This key limitation meant that the perceptron could only classify patterns that were linearly separable. Back-propagation overcomes this limitation because it can adapt two or more layers of weights, and uses a more sophisticated learning rule. The power of back-propagation lies in its ability to train hidden layers which act as layers of *"feature detectors"* - units that respond to specific features in the input pattern, and thereby escape the restricted capabilities of single-layer networks.

Typically, back-propagation employs three or more layers of processing units. Figure 5-5 shows the topology for a typical three-layer back-propagation network. The bottom layer is the input layer - the only nodes in the network that receive external input. The above is the hidden layer, in which the processing units are interconnected to layers above and below. The top layer is the output layer. Back-propagation networks do not have to be fully interconnected, although most applications have been done with fully interconnected layers.

A back-propagation neural network is trained by supervised learning. The network is

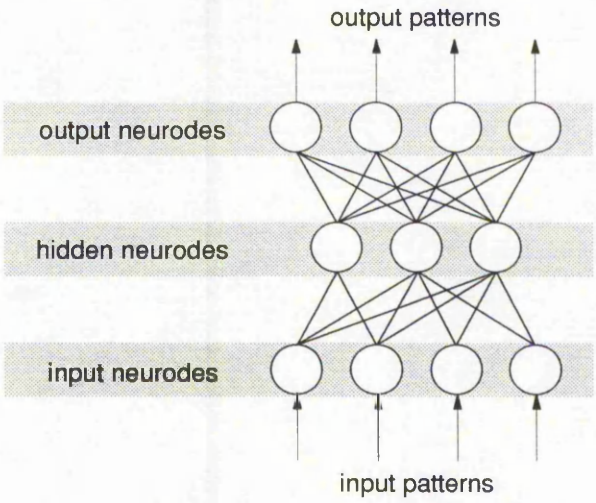


Figure 5-5: *A three-layered back-propagation network (fully interconnected)*

presented with pairs of patterns - a set of input pattern together with a target output(s). Upon each presentation, weights are adjusted to decrease the difference between the network's output and target output. A training set (a set of input / target pattern pairs) is used for training, and is presented to the network many times. After training is stopped, the performance of the network is tested.

The back-propagation learning (training) algorithm involves a forward-propagating step followed by a backward-propagating step. The forward-propagation step begins with the presentation of an input pattern to the input layer of the network, and continuous as activation level calculations propagate forward through the hidden layers. In each successive layer, every processing unit sums its inputs and then applies a *sigmoid function*, which is shown in Figure 5-6, to compute its output. The output layer of units then produces the output of the network. The following assumes a sigmoid logistic non-linearity is used where the function $f(x)$ in Figure 5-6 is

$$f(x) = \frac{1}{1+e^{-x}} \tag{5-1}$$

The back-propagation training algorithm is a generalisation of the *least mean square (LMS)* algorithm. It uses an iterative gradient search technique to minimise a cost function equal to the mean square error between the actual output of a multilayer feed-forward perceptron and the desired output [DAY90] [LIP87]. It requires continuous differentiable non-linearities.

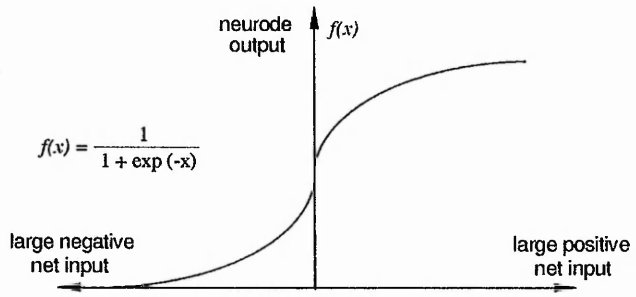


Figure 5-6: *Sigmoid transfer function used in back-propagation network*

Notation	Description
i	input value
o	output value
t	target value
w	connection weight
n	number of nodes in a layer
i (subscript)	input layer
h / j (subscript)	hidden layer
l (subscript)	output layer
p (subscript)	given pattern
η	learning coefficient (eta)
α	momentum factor (alpha)
δ	error term (delta)

Table 5-1: *Notation of the back-propagation Network*

A *bias unit* is employed as part of every layer but the output layer of the back-propagation network. This unit has a constant activation value of 1. Each bias unit is connected to all units in the next higher layer, and its weights to them are adjusted during the back-error propagation. The bias units provide constant term in the weights' sum of the units in the next layer. The result sometimes is an improvement on the convergence properties of the network.

Table 5-1 illustrates the notation chosen by the author which is going to be used throughout the document for back-propagation networks. Just as there is no standard for back-propagation network implementation, there is no standard for its notation. The author does hope that some clear, consistent standard notation is adopted soon. The detailed implementation of the network is discussed next.

5.2.3 Implementation

To describe the implementation of the back-error propagation model, we first look at the ways that input is presented to a network and the normalisation techniques used. Next

we present the equations that describe the network training and operation. These equations are divided into two categories: feedforward calculations and error-propagation calculations. The feedforward calculations are used both in training mode and in the operation of the trained network. Back-propagation calculations are applied only during training.

5.2.3.1 Network input

On the left of Figure 5-3, inputs are shown coming into the input layer of the network, to a layer of processing neurodes. Beware of the urge to "mix and match" the input data in an attempt to reduce the number of input neurodes. For instance, resist the urge to combine parameters before presentation to a neurode. It will be more efficient for the engineer to allow the network to take a little longer to train successfully, than if it fails to train at all.

For the version of the back-propagation network created by the author, each input can take on any value between zero and one. Does this *normalisation process* constrain the engineers in any way ? Probably not. Whenever we deal with a digital computer system that is receiving input, we are limited by the size of the number we can put in. As long as the resolution of the input data does not get lost in the normalisation process, we are all right.

To implement the network described here, the source codes written in C use standard floating point variables. This type of variable is 32 bits in length and uses 24 bits for the value and 8 bits for the exponent. We, therefore, have a resolution of about one part in sixteen million, or seven decimal places. Normalising input patterns can actually provide a tool for pre-processing data in different ways. The pattern can be normalised by considering all of the n inputs together, normalising each input channel separately, or normalising groups of channels in some way that makes sense. In some cases, the way chosen to normalise the inputs can affect the performance of the network, so this is one place to try different approaches.

5.2.3.2 Feedforward calculations

In the following section, mathematical equations that describe the training and testing / running modes of a back-propagation network are presented. They are presented without derivations or proofs. This information can be found in Rumelhart and McClelland [RUM86], specifically in Chapter 8.

The net input to a hidden neurode is calculated as the sum of the values for all connections coming into the neurode, as described in Equation 5-2 (refer to Table 5-1 for the notation). Note that this includes the input from the neurode we call the bias unit, which is assumed to have an output of one at all time and is otherwise treated as any other neurode.

$$input_j = i_j = \sum_i w_{ji} o_i \tag{5-2}$$

$$output_j = o_j = \frac{1}{1+e^{-i_j}} \tag{5-3}$$

The output of a hidden neurode as a function of its net input is described in Equation 5-3. This is the sigmoid function to which we previously referred. The nonlinear nature of this sigmoid transfer function plays an important role in the performance of the neural network. Other functions can be used as long as they are continuous and possess a derivative at all points. Once the outputs of all hidden layer neurodes have been calculated, the net input to each output layer neurode is calculated in an analogous manner, as described by Equation 5-4. Similarly, the output of each output layer neurode is calculated as described by Equation 5-5.

$$input_l = i_l = \sum_j w_{lj} o_j \tag{5-4}$$

$$output_l = o_l = \frac{1}{1+e^{-i_l}} \tag{5-5}$$

During the feedforward calculations, two mathematical operations are performed by each neurode, and the output state, or activation, is obtained as a result. The first is a summation of previous layer neurode outputs multiplied by interconnecting weights, and the second is the squashing function (whose value is always between finite limits, even the input is unbounded).

5.2.3.3 Training process

During the training phase, the feedforward output state calculation is combined with backward error propagation and weight adjustment calculations that represent the network's learning, or training. Equation 5-6 presents the *definition of the error.* Note that the engineer implements the error calculation in the back-propagation training algorithm on a neurode-by-neurode basis over the entire set of patterns, rather than on a pattern-by-pattern basis. The errors have been summed over all neurodes, giving a grand total for all neurodes and all patterns. We, then, divided the grand total by the number of patterns, to give an *average sum-squared error value.*

$$Error_p = 0.5 \sum_{i=1}^{n_l} (t_{pl} - o_{pl})^2 \qquad (5\text{-}6)$$

The goal of the training process is to minimise this average sum-squared error over all training patterns. The error signal is represented by δ_l for output layer neurodes and defined by Equation 5-7 (in the case of using the sigmoid function).

$$\delta_l = o_l(1 - o_l)(t_l - o_l) \qquad (5\text{-}7)$$

We implement the back-propagation algorithms using batch (epoch) training which accumulates the δ'*s* for each neurode for the entire training set, adds them, and back propagates the error based on the grand total δ. Before we can update weights, we initialise each weight to some value. Neural network researchers have recommended a number of

variations on the initial weight range. For example, Lee [LEE89] has shown that in some instances initialising the weights feeding the output layer to random values between -0.3 and 0.3, while initialising weights feeding the hidden layer to 0. In the experiments, the random number initialises to values from -0.3 to 0.3 works well and is almost always a good place to start.

Now we look at how to update weights that feed the output layer w_{lj} by using δ_l. To a first approximation, the updating of these weights is described by Equation 5-8. The learning coefficient (η) can be assigned values between 0 and 1.

$$w_{lj}^{new} = w_{lj}^{old} + \eta \delta_l o_j \tag{5-8}$$

This kind of weight updating sometimes gets caught in what are called *local energy minima* [DAY90][EBE90]. Ideally, we would like to move the position to the bottom of the bowl where the energy is minimum; this position is called the *globally optimal solution.* We can help the search of global minima by using the *"momentum"* factor which can take on values between 0 and 1. Equation 5-9, which is just Equation 5-8 with the momentum term added, becomes the equation the engineer actually uses in the back-propagation network to update the weights feeding the output layer.

$$w_{lj}^{new} = \Delta w_{lj}^{old} \alpha + w_{lj}^{old} + \eta \delta_l o_j \tag{5-9}$$

where Δw is the previous weight change (the new weight is equal to the old weight plus the weight change).

Now that we have the new values for the weights feeding the output neurodes, we turn our attention to the hidden neurodes. Again the author refers to the derivation by Rumelhart and McClelland [RUM86]. They show that the error term for the hidden neurode is presented by Equation 5-10. The weight changes for the connections feeding the hidden

layer from the input layer are now calculated in a manner analogous to those feeding the output layer which is described by Equation 5-11 (refer to Table 5-1 for the notation).

$$\delta_h = o_h(1-o_h)\sum_{l=0}^{n_l} w_{lh}\delta_l \tag{5-10}$$

$$w_{ji}^{new} = \Delta w_{ji}^{old}\alpha + w_{ji}^{old} + \eta\delta_j o_i \tag{5-11}$$

The choices of η and α depend on the applications. Rumelhart and McClelland [RUM86] frequently use values of 0.5 and 0.9, respectively, as a place to start.

5.2.3.4 Testing and running the network

The set of calculations that results in obtaining the output state of the network, which is simply the set of the output states of all of the output neurodes, is carried out in exactly the same way during the training phase as during the testing / running phase. The test / run operational mode just involves presenting an input set to the input neurodes and calculating the resulting output state in one forward pass.

5.2.4 Software engine

In this section, the author takes a detailed look at how the back-propagation network is implemented using C programming language, specify and run the network on PCs, and examine how a network specification is turned into working code. Figure 5-7 shows the actions of a single processing element. Each processing element computes its output or activation as a function of its inputs. Inputs are weighted and summed to form the net input to the element.

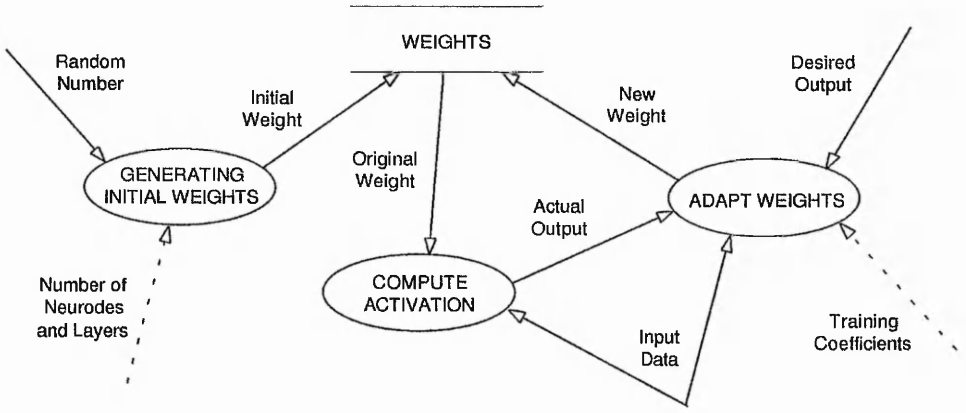


Figure 5-7: *DFD for processing element data model*

5.2.4.1 Storage allocation

In most neural network implementations, it is necessary to manipulate large vectors and matrices of numbers for recording the massively parallel architecture. Activation values and weights are stored and manipulated as vectors, which are especially important when co-processors are used because engineers can get a much better performance by giving them whole vectors to compute all at once, rather than element by element. For many real-world applications, the storage needed for the data can become substantial because there are lots of processing elements and even more connections. Consider for example, a three-layer network with 30 input nodes, 36 hidden nodes, and 5 output nodes. Assuming 200 patterns are used for training, we get :

$$\begin{aligned} \textit{Number of bytes of storage} &= (\text{Number}^{\textit{activation values}} + \text{Number}^{\textit{interconnections}}) \times 8 \\ &= \{[200 \times (30 + 36 + 5)] + [5 \times (36 + 1) + 36 \times (30 + 1)]\} \times 8 \\ &= (14200 + 1301) \times 8 \\ &= 124008 \textit{ (bytes)} \cong 121 \text{ Kbytes} \end{aligned}$$

The amount of storage needed to run a neural network is governed by the size of the executable code plus the size of the network. To up speed training, patterns are read in once from the input device, e.g. hard disk driver, and store in memory. An alternative, *much*

slower, method is to repeatedly read the patterns from disk for each presentation to the network. Clearly, storing all patterns simultaneously in memory is a faster method but it demands more storage. On the other hand, keeping patterns on disk is slower but uses less memory space. On machines with 32-bit addressing, such as the 80386 or 80486, it is possible to use more sophisticated virtual memory techniques to run the network.

The simplest way to declare for activation values and weights is to use *static storage*. The problem of this simple scheme is that neural networks are memory hungry : 121 Kbytes were needed for weights and activation values for the example given earlier. This does not work on MS-DOS based systems because of the 64-Kbytes segment limit by the processor architecture. Dynamic arrays use storage most effectively and solve the problem encountered with static arrays. Dynamic array sizes can be available and limited only the amount of physical memory available. A vector is a one-dimensional array, and a matrix is two dimensional. The C language does not directly support two-dimensional memory allocations. A matrix is declared as an array, with each element being an array [EBE90]. Figure 5-8 presents how to visualise a two-dimensional matrix (15 ×10). The element in the x^{th} row and y^{th} column is referred to by *matrix*[*x*][*y*], just as it would be for the static array. Therefore, once storage has been allocated, reference to the array need not be aware of the particular storage allocation technique used. This makes for easy transparent use in

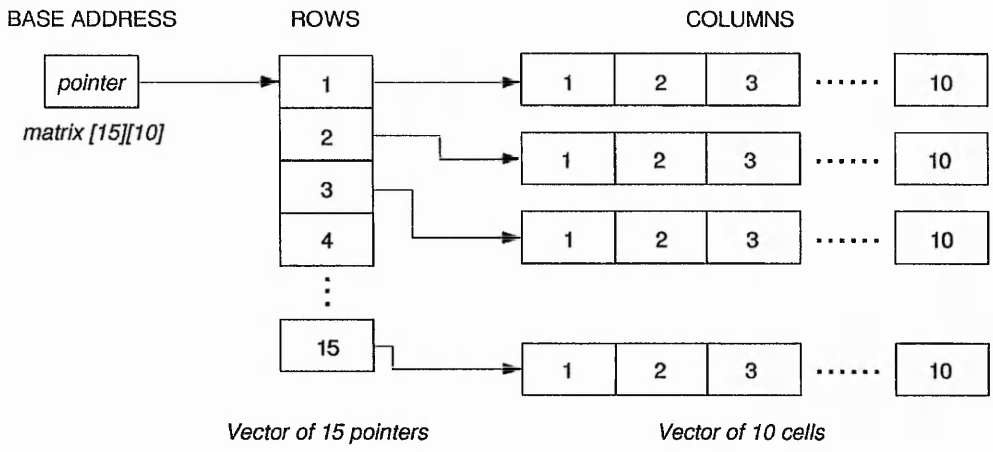


Figure 5-8: *Dynamic memory allocation of a two-dimensional matrix*

programs. Next, the author is going to describe how a network specification is turned into working codes in the following sections.

5.2.4.2 Propagating error signals

The error term at the output layer, *delta_out*, is computed from the difference between the actual output and the desired *target* values for each node in the output layer, for each training pattern:

$$delta\_out = (target - vector\_out) \times vector\_out \times (1 - vector\_out)$$

where *vector_out* is the activation vector at the output layer. The error at the hidden layer, *delta_hidden*, is calculated recursively from the error terms of units in the output layer, using the following formulas:

$$sum = sum + delta\_out \times weight(hidden\text{-}output)$$
$$delta\_hidden = sum \times vector\_hidden \times (1 - vector\_hidden)$$

where *delta_out* x *weight(hidden-output)* is the product of the delta of an output unit and the weight of the link between the hidden unit and the output unit. Also *sum* is the error term, derived from all output units to which the hidden unit is connected.

5.2.4.3 Adapting weights

The weight changes depend on the propagated error terms. The magnitude of the weight change is controlled by the learning rate constant *eta* (η). The higher eta is the bigger the weight changes and the faster the network is able to train. Nevertheless, high eta increases the chance of oscillation. The momentum term *alpha* (α) damps high-frequency

weight changes and reduces the risk of oscillation while still permitting fast learning rates. The *change in weight* at the hidden to output connections is:

$$weight\_change = [(eta \times (delta\_out \times vector\_hidden)] + (alpha \times old\_delta\_out)$$

This product is summed over all units of the hidden layer. The new value of the weight is:

$$weight(hidden\text{-}output) = weight(hidden\text{-}output) + weight\_change$$

Note that the *bias* weight (Section 5.2.2.2) is calculated in exactly the same way as any other weights. The adaptation of the input to hidden layer weights follows a similar set of rules. This completes the description of the engine for a back-propagation neural network.

5.2.4.4 Training successfully ?

How does an engineer know whether the network is training correctly, and how can an engineer tell how well it is doing? The easy way is to look at the output nodes and compare them to the target values. The difference between the actual output and the target gives the error at a node for a given pattern:

$$error = error + (target[p][j] - vector\_out[p][j])^2$$

By summing the error over all output nodes for all patterns and taking the average, we get a measure of training performance. The *average sum squared error* is presented by the following formula. When the error is less than the required maximum error (or satisfactory error level), training is terminated. And the network is ready for use.

$$error = error \div (number\_patterns \times number\_output\_nodes)$$

5.3 Neural Fuzzy Theory

In recent years, the studies on *neural fuzzy systems* have been developed on the basis of theories of neural networks and fuzzy set. Human reasoning is somewhat fuzzy in nature, so the typical neural fuzzy systems should process some reasoning ability. Additionally, a neural fuzzy system should be designed to implement the fuzzy implication operation, which is considered to conform to the characteristic of dynamic thinking action of human beings, and take advantages of the fuzzy approximation inference theory conveniently. In the following sections, the author has discussed the concepts of neural fuzzy systems and presented an elegant scheme to integrate the neural network and fuzzy logic.

5.3.1 Introduction

The controls of complex systems or processes have been successfully treated by many control engineers and theorists. However, the complexity of their analytical models makes the analysis of their input/output relation and their analytical approach very difficult. Hence, many researchers and engineers have focused their attention on artificial neural networks and fuzzy control systems, where the applications are frequent and give good results [LEE90] [MAM74] [RUM86], as viable alternatives to traditional analytical modelling and control. These applications have successfully been applied to control dynamic systems for which the conventional control methods do not always satisfy all the requirements.

Since most machining processes are stochastic, nonlinear and sometimes unclearly defined, they are suited to control by means of fuzzy logic. Materials cutting and processing in general are good examples of these processes. Numerous applications of the fuzzy systems to the control of ill-defined complex process have been reported since Mamdani's first paper [MAM74]. Conventional and modern control theories need a precise knowledge of model of the process to be controlled and exact measurements of input and output

parameters. However, due to the complexity and vagueness of practical processes, the application of these theories is still limited.

Fuzzy logic based design has several advantages including simplicity and ease in design. However, fuzzy logic design is associated with some critical problems as well. As the system complexity increases, it becomes difficult to determine the right set of rules and membership functions to describe the system behaviour. A significant amount of time is needed to properly tune the membership functions and adjust rules before a solution is obtained. For more complex systems, it may be even impossible to come up with a working set of rules and membership functions. Besides, once the rules and membership functions are determined, they remain fixed in the fuzzy logic controller, i.e. controller cannot learn from experience.

Artificial neural networks have been shown to have the capability of approximating arbitrarily complicated continuous functions. The training process of the artificial neural network reduces the overall network error to a minimum by adapting its weights based upon the error function. The technology of neural networks has been successfully applied to solve many difficult engineering problems such as control [KUM90] [NAR90], fault detection and pattern recognition [CHO93b] [VIL90], among others. Nevertheless, one of the drawbacks of using conventional neural network technology alone to solve engineering problems is its *"black-box"* characteristics. In most cases, engineers cannot interpret the network's decision-making process from a heuristic point of view; they can only know that the network gives a correct input-output mapping. Clearly, neural networks can solve many complex problems. But it may not be the most effective way to implement it, as today, implementation of neural network is more costly compared to fuzzy logic implementation. A conventional embedded controller can be easily used to implement fuzzy logic by proper programming. Neural implementation by programming is also possible but will be slower. A dedicated hardware implementation is also more common today for fuzzy logic than neural networks, especially considering cost.

Use of neural networks associated with fuzzy logic based designs to learn system behaviour seems to be a good way to solve above mentioned problems. Using system's input-output data, neural networks can learn the behaviour of the system and accordingly can generate fuzzy rules and membership functions. Neural networks is a data driven system and does no use programming. By proper learning, neural net can develop good generalisation capabilities and thus, can solve many problems that are either unsolved or inefficiently solved by existing techniques.

5.3.2 Principles

In many real processes, control relies heavily upon human experience. Skilled human operators can control such processes quite successfully without any quantitative models in mind. The control strategy of the human operator is mainly based on linguistic qualitative knowledge concerning the behaviour of an ill-defined process. Fuzzy controllers have been implemented in many types of systems with success. It is found to have the desired robustness quality for control under disturbances and plant uncertainties. Knowledge in linguistic form can be coded as rules in a fuzzy controller, thus enabling the use of knowledge from an expert operator in the design.

It has been observed that many real problems, such as pattern recognition and control, are solved effectively by artificial neural networks. This is due to their non-linear mechanism, and their self-learning capability. Because neural networks are designed in an attempt to mimic the human brain, they emulate human performance and thereby function intelligently. On the other hand, fuzzy set theory, which can model uncertainty or ambiguous data so often encountered in real life, enable a system to tackle real life situations in a manner more like humans, which is noted to be somewhat fuzzy in nature. Neural networks and fuzzy logic have been used for systems decision and adaptation. Neural networks have provided a robust means of making systems decisions for nonlinear applications, while fuzzy logic has proven capable of properly classifying "gray area" decisions.

The key benefit of fuzzy logic is that it lets you describe the desired system behaviour with simple *"if-then"* relations. In many applications, this gets an engineer a simpler solution in less design time. It can also be used to optimse the performance directly. While this is certainly the beauty of fuzzy logic, at the same time it is its major limitation. In some cases, the knowledge that describes desired system behaviour is contained in data sets. Here, the designer has to derive the *"if-then"* rules from the data sets manually, which imposes a major effort with large data sets. Most people have hence looked into neural networks. However, only in a few applications have neural net solutions shown better results compared with other methods.

The recent growth in attention to neural networks on the one hand and fuzzy logic on the other has led to many suggestions for their combined use in control. Many simulation programmes and actual hardware implementations of systems are now available which demonstrate that these techniques can actually work in applied domains. Similarities exist between the neural networks and the fuzzy logic controllers. They both can handle extreme nonlinearities in the system. Both techniques allow interpolative reasoning which frees us from the true / false restriction of logical systems such as the ones used in symbolic AI. For instance, once a neural network has been trained for a set of data, it can interpolate and produce answers for the cases not present in the training data set. Similar properties hold for a fuzzy controller. The weighted average scheme of fuzzy control and the sum of the products of the neural networks are similar in principle. It has been shown that both of these techniques can use interpolative reasoning which enables them to go beyond the traditional true-false restriction of the AI symbolic methods.

To put it briefly, both neural networks and fuzzy logic are powerful design techniques which both have their strengths and weaknesses. Neural networks can learn from data sets, while fuzzy logic solutions are easy to verify and optimse. Table 5-2 presents a summary of these properties. It becomes obvious that a clever combination of the two technologies

	Fuzzy Logic	Neural Networks
Knowledge representation	Explicit, verification and optimisation are easy and efficient (+++).	Implicit, the system cannot be easily interpreted or modified (–).
Trainability	None - everything must be explicitly defined.	Trains itself by learning from examples - data sets (+++).

Table 5-2: *The strengths and weaknesses of fuzzy logic and neural networks*

delivers the best of both worlds [ALT94]. Combine the explicit knowledge representation of fuzzy logic with the learning power of neural networks, and we get *Neural Fuzzy* technologies.

Synthesis of neural network with fuzzy logic offers a key advantage over traditional control systems. They offer model-free estimation of a control system. The user need not specify how the controller's output mathematically depends on its input. Instead fuzzy methods model the control system using fuzzy inference rules to construct the fuzzy controller. In this case, the user only needs to provide structured knowledge of the control process. In addition, neural approach may be preferable if the user can provide a statistically representative set of numerical training samples of the system. Because of these advantages, we consider in incorporating the concept of fuzzy set into the neural networks to construct a *neural fuzzy system.* The main idea in integrating the fuzzy logic controllers with neural networks is to use the strength of each one collectively in the resulting neural fuzzy system. This fusion allows:

1) A human understandable expression of the knowledge used in control in terms of the fuzzy control rules. This reduces the difficulties in describing the trained neural network which is usually treated as a black box;
2) The fuzzy controller learns to adjust its performance automatically using a neural network structure and hence learns by accumulating experience.

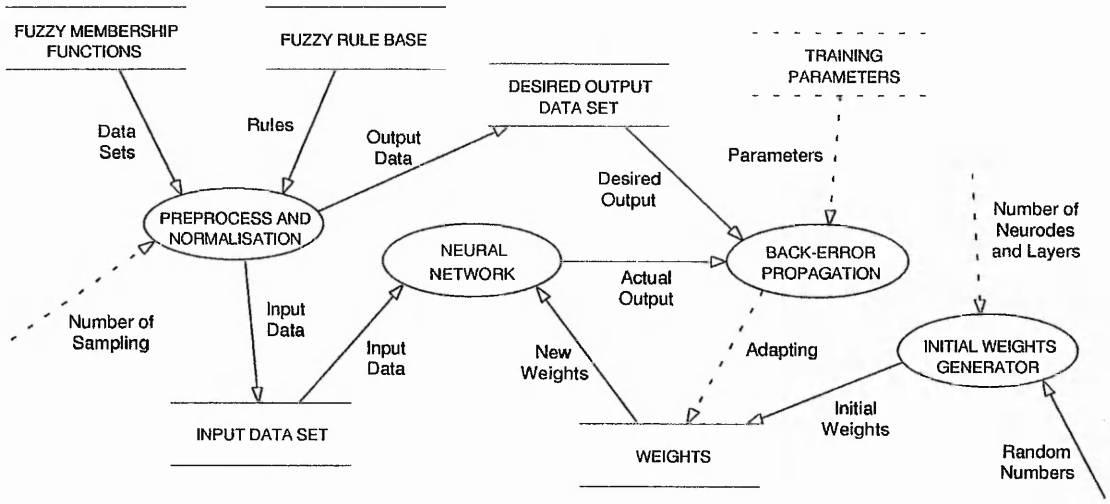


Figure 5-9: *General architecture of neural fuzzy systems*

It is noted that although fuzzy logic is a mechanism for propagating uncertainty, it may involve in some cases an increase in the amount of computation with high flexibility.

Figure 5-9 presents a general architecture of the proposed neural fuzzy system. This system reads desired fuzzy input/output membership functions and fuzzy rule base, after pre-processing and normalising these data, the actual training sets (a set of input/output pattern pairs) for the neural network can be generated, and is presented to the network many times. The initial weight for the connection between neurodes in the different layers is produced by a weight generator which is initially randomised with values from within an interval, e.g. between -0.3 and 0.3. A multi-layer feedforward network with sigmoid elements is employed, and the back-propagation supervised training algorithm is used for training. After training is complete, the network can be used for imitating the inference procedures of the fuzzy system.

The concept of *neural fuzzy system* has recently received much attention [LIN91] [MIT92] [OKA92]. Many alternative ways of integrating neural networks and fuzzy logic have been proposed in scientific literature [CEL92] [CHO93a] [FRE93] [KHA93] [LEN93]. Only very few have been already successfully applied in industrial applications. In the

following sections the author is going to describe a novel method used to construct a modified version of *neural fuzzy kernel* which has been successfully implemented and tested in this project for solving complex problems. Before we move on this topic, first, we would like to discuss the responses of fuzzy systems. There are mainly three types of fuzzy membership functions commonly used at present - symmetric triangles, symmetric trapezoids and "bell" shaped functions. By associating with different shapes of fuzzy input/output membership functions, refer to Figure 5-10, the system can produce various kinds of responses for different applications. In the following section, the reasons for combining symmetric triangles and symmetric trapezoids to define the system input and output membership functions for constructing the neural fuzzy system used in this project are explained.

5.3.3 Fuzzy inferences

In this section, two experimental results have been presented to explicate the characteristic of the fuzzy responses. Different combinations of the input / output fuzzy membership functions have been tested.

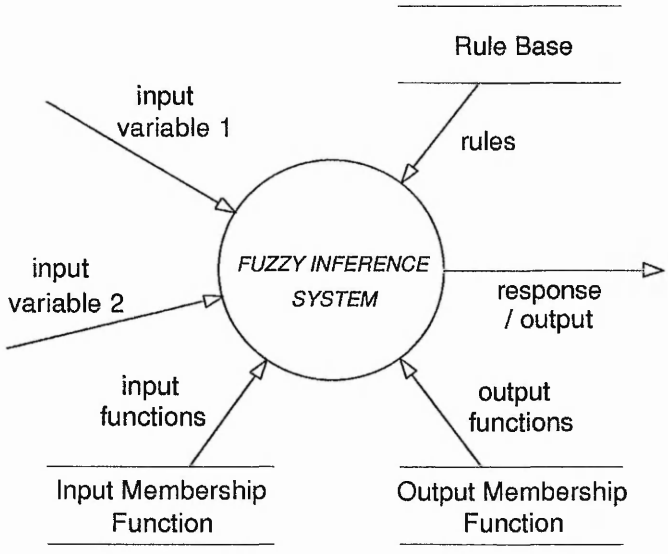


Figure 5-10: *Overview of a fuzzy inference engine*

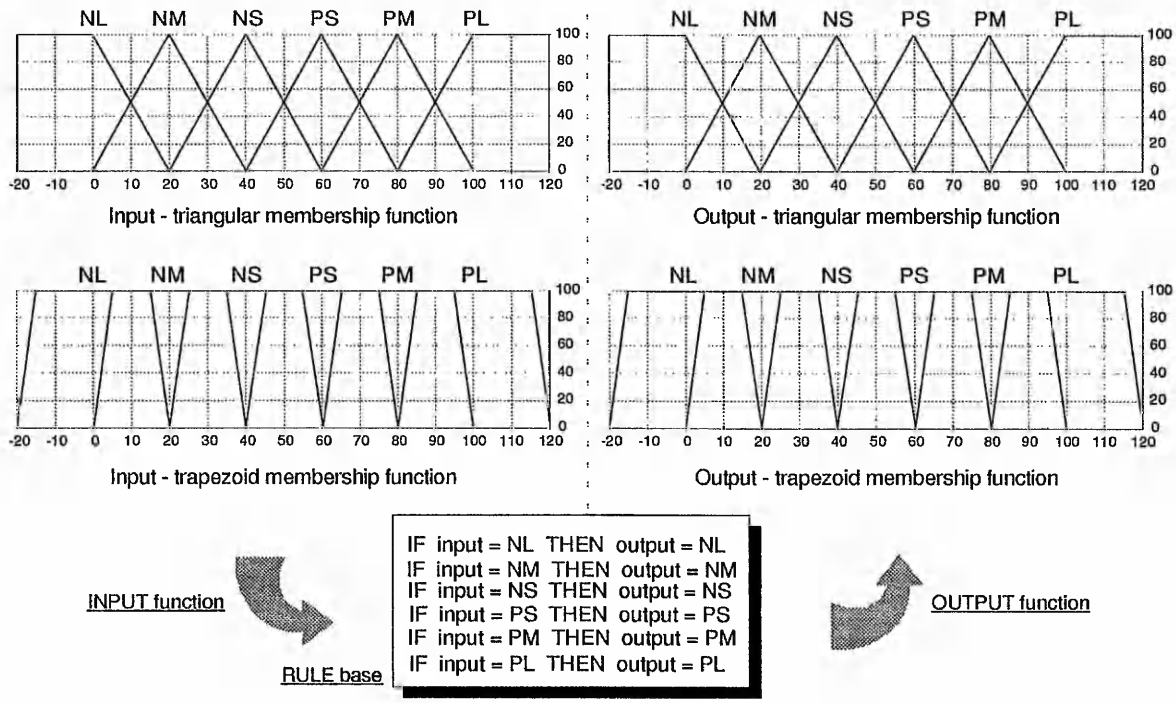


Figure 5-11: *Fuzzy membership functions and the rule base*

5.3.3.1 One input with one output

Two different types (symmetric triangles and symmetric trapezoids) of fuzzy membership functions as well as the rule base, which have been applied in this experiment, are illustrated in Figure 5-11. Both input and output functions are separated into six overlapped fuzzy sets (NL, NM, NS, PS, PM and PL, where N is Negative, P is Positive, L is Large, M is Medium and L is Large). Using combinations of these two types of membership functions, four different system response patterns can be obtained from the output of the engine.

Figure 5-12 shows the system response patterns acquired from the one-input and one-output fuzzy engine. It is obvious to observe that the use of the symmetric triangle input and symmetric trapezoid output membership functions in the fuzzy engine can produce the best (closest) outputs to the target pattern.

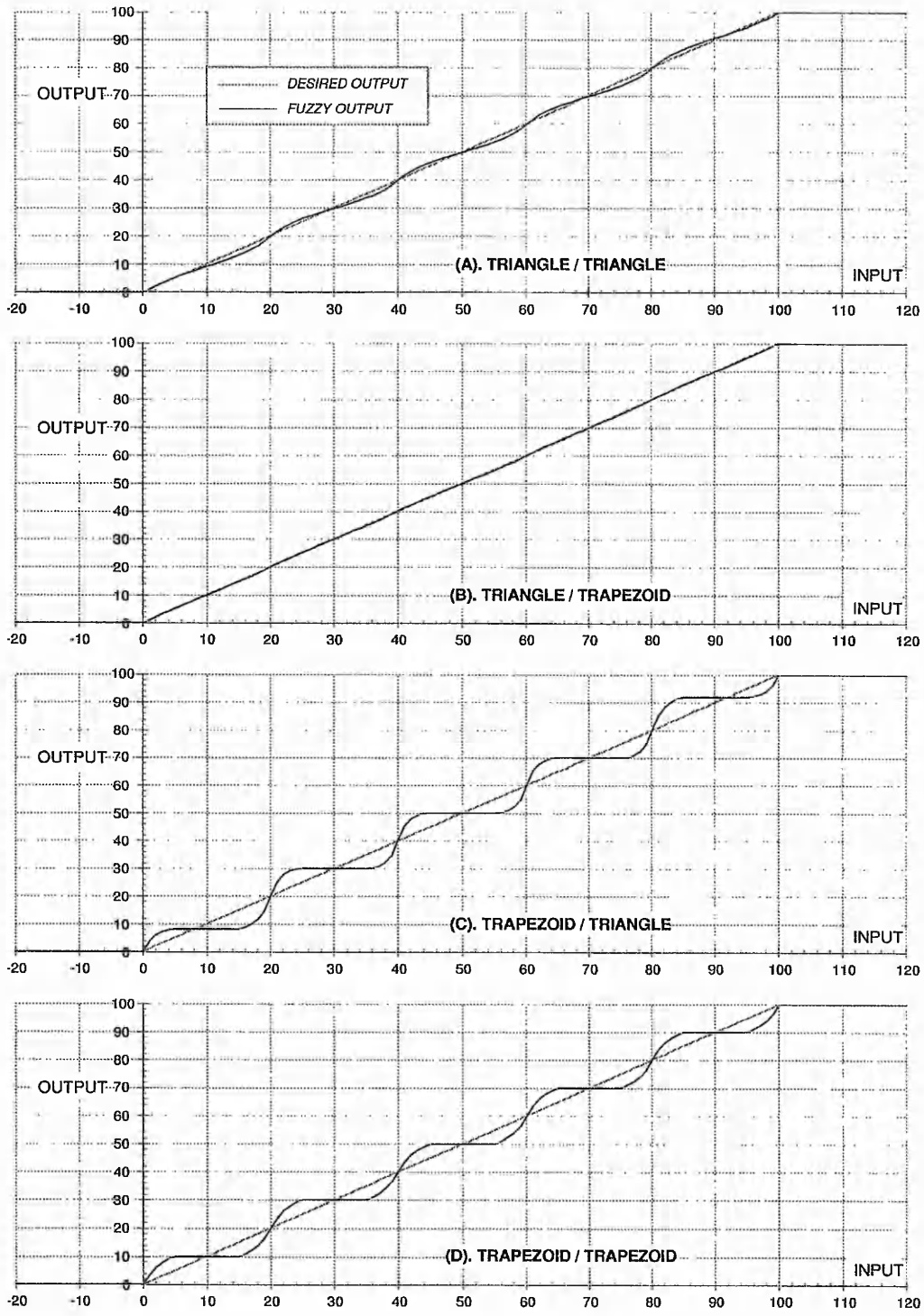


Figure 5-12: *Responses of the fuzzy engine with one input and one output*

5.3.3.2 Two inputs with one output

The fuzzy system using two-inputs and one-output is one of the most common utilised inference engines. Similar to Figure 5-11, symmetric triangles and symmetric trapezoids are

γ		β				
		NL	*NS*	*MD*	*PS*	*PL*
α	*NL*	NL	NM	NS	MD	PS
	NS	NS	NM	MD	PS	PS
	MD	MD	NS	MD	PS	PM
	PS	PS	PS	PS	PM	PM
	PL	PM	PM	PM	PL	PL

Table 5-3: *Fuzzy rule base, FAM Bank*

applied in the testing. The input membership function is divided by means of five overlapped fuzzy variables (fuzzy sets). Also the output membership function is separated into seven regions. Twenty-five rules are used in the inference process. The Fuzzy Associative Memory (FAM) Bank (refer to Section 4.3.2.3) is depicted in Table 5-3.

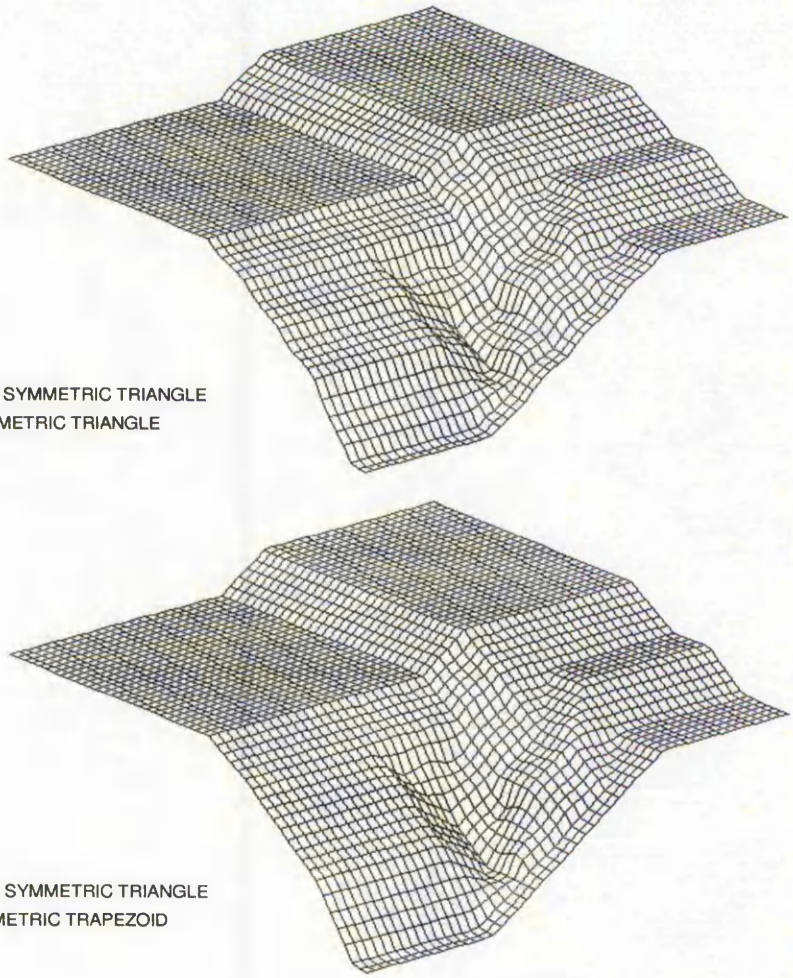


Figure 5-13(a): *Responses of the fuzzy system with two inputs and one output*

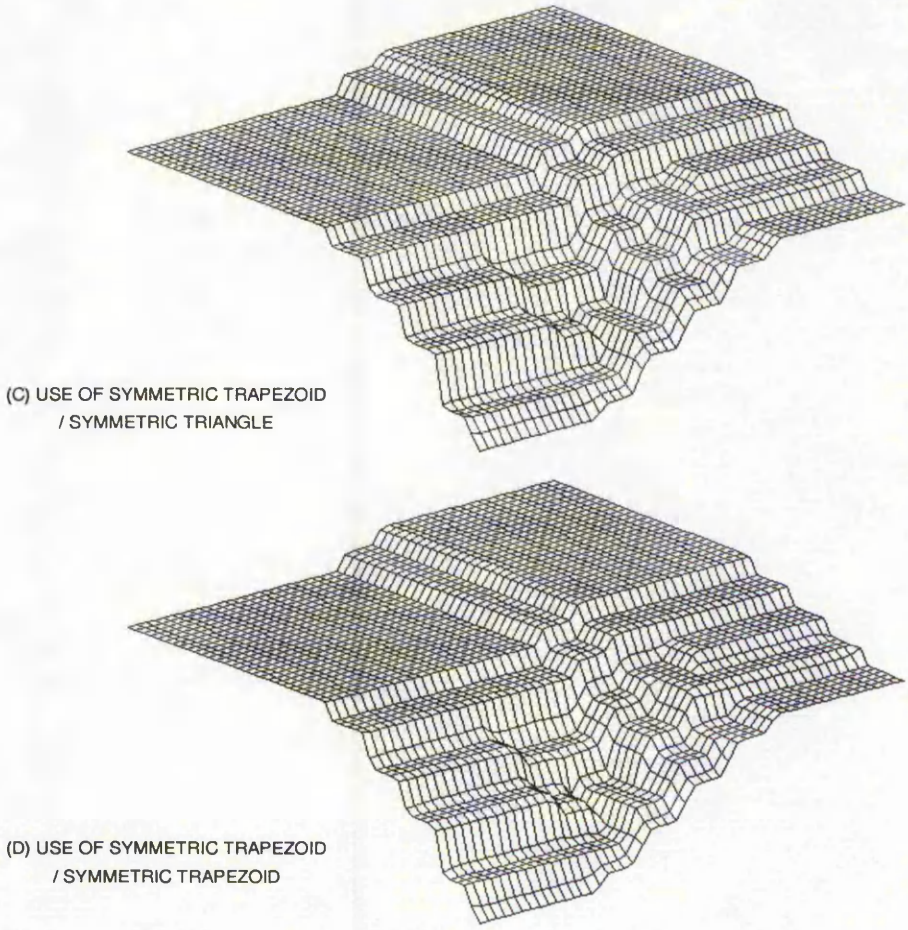


Figure 5-13(b): *Responses of the fuzzy system with two inputs and one output*

The fuzzy engine reads two external values (α and β) to assign degrees of membership in one or more fuzzy sets. Strengths are computed based on the rule base and presented to the rules' fuzzy output (γ). Finally, the *"centre-of-gravity method"* is employed to calculate the average weight for system output. Figures 5-13 (a) and (b) illustrate the fuzzy outputs using different combinations of membership functions.

5.3.3.3 Discussion

It is obvious to see (Figure 5-12) that the output patterns of the fuzzy engine can be changed dramatically in terms of associating two types of fuzzy membership functions. The use of the symmetric triangle input and symmetric trapezoid output membership functions

(type (B) labelled in Figure 5-12) in the fuzzy engine can provide the closest outputs to the desired pattern. Careful inspection of Figure 13 (a) and (b) reveals that the inference engine using triangular and trapezoid membership functions can produce the smoothest output pattern comparing to other combinations. In this project, this type of membership functions are used to build the inference engines.

5.3.4 Construction of neural fuzzy system

5.3.4.1 Problems of current approaches

The concept of neural fuzzy theory has received much attention recently. Various alternative ways of integrating neural networks and fuzzy logic have been proposed in the scientific literature [CEL92] [CHO93a] [FRE93] [KHA93] [LEN93]. The problem of using current neural fuzzy techniques in the proposed project is that the shape of the fuzzy membership functions generated by these methods do not ssatisfy the requirements of the proposed system. As mentioned in the previous section, in order to obtain an ideal response pattern the symmetric triangular and trapezoid membership functions are employed for the construction of the inference engine. Using a similar method described in [LEN93], the system can produce the membership function as depicted in Figure 5-14.

The membership function of this fuzzy system is implemented using multi-layer pre-trained neural network which enables off-line learning of the function using *fully connected* back-propagation feedforward network. Diagram (a), (b) and (c) in Figure 5-14 show the final results of the membership functions used 0.001, 0.000001 and 10^{-12} as the satisfactory error (δ) level (or error tolerance) to train the neural fuzzy system, respectively. It is clear from the figure that the response pattern (actual output) cannot entirely match the desired output. The problem of using such membership function is that these imprecise output values will cause a certain amount of error and interfere with the performance of the neural fuzzy engine, and further affect the entire system. According to the obtained experimental

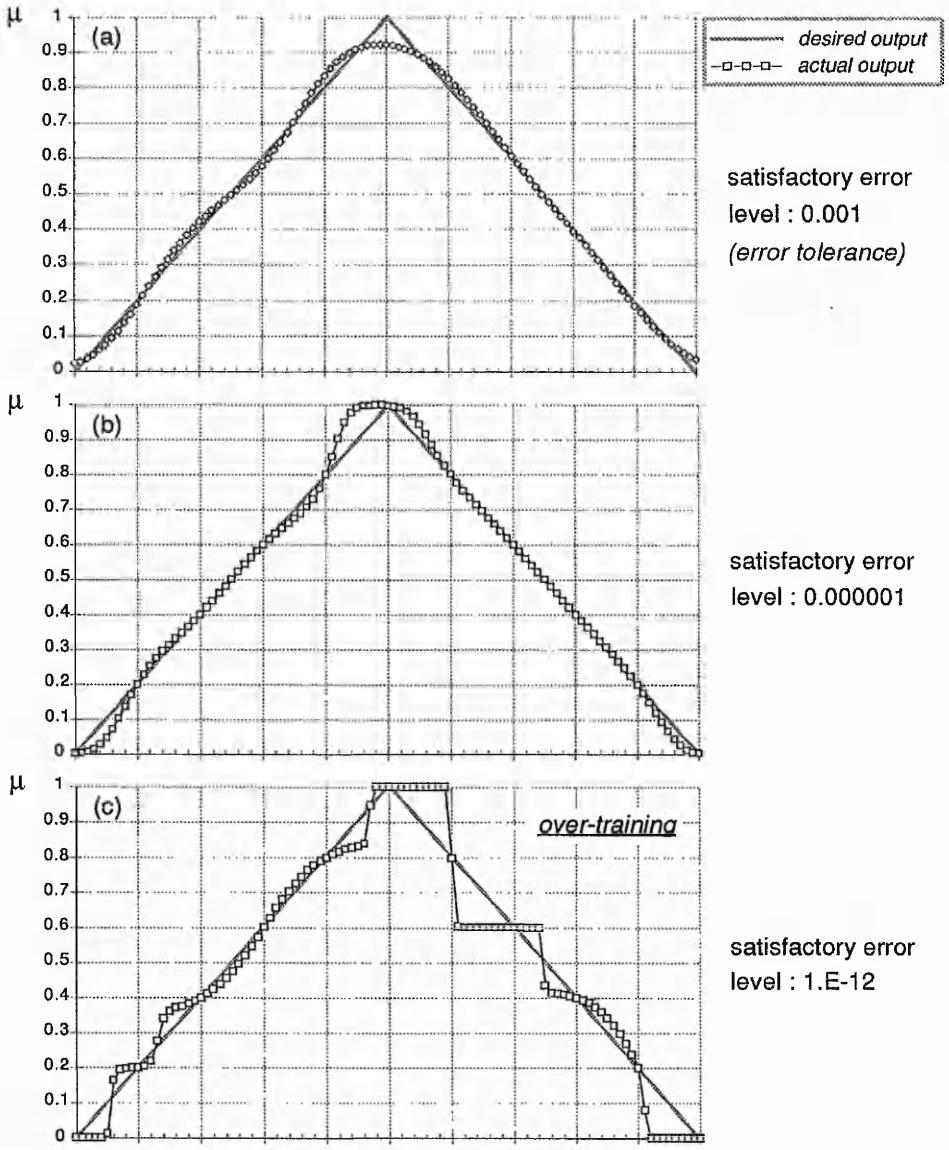


Figure 5-14: *Example of using common neural fuzzy technique to generate the fuzzy membership function*

results, such as shown in Figure 5-14, it seems impossible to make an ideal triangular membership function by utilising this common neural fuzzy architecture.

5.3.4.2 Proposed neural fuzzy architecture

In order to generate suitable fuzzy membership functions suitable to the project, various kinds of neural network architectures have been investigated and tested. Two different

structures for generating the *system input membership functions* which produced superb results are described in detail next. The methods used to produce the *system output membership functions*, and transfer the *fuzzy rule base* into a set of network connections are also presented.

5.3.4.2.1 Generating fuzzy input membership functions

In order to design an effective architecture for training the neural fuzzy system suitable to the problem in hand, several initial experiments were undertaken to construct the membership function. A small number of representative data set were used for these experiments. As a result of these experiments, two types of network have been found to have excellent effects. These novel network architectures are described below.

(1) Type A

The first type of neural network architecture used is fully connected multi-layer network with sigmoid activation function, five neurodes in the first hidden layer and two

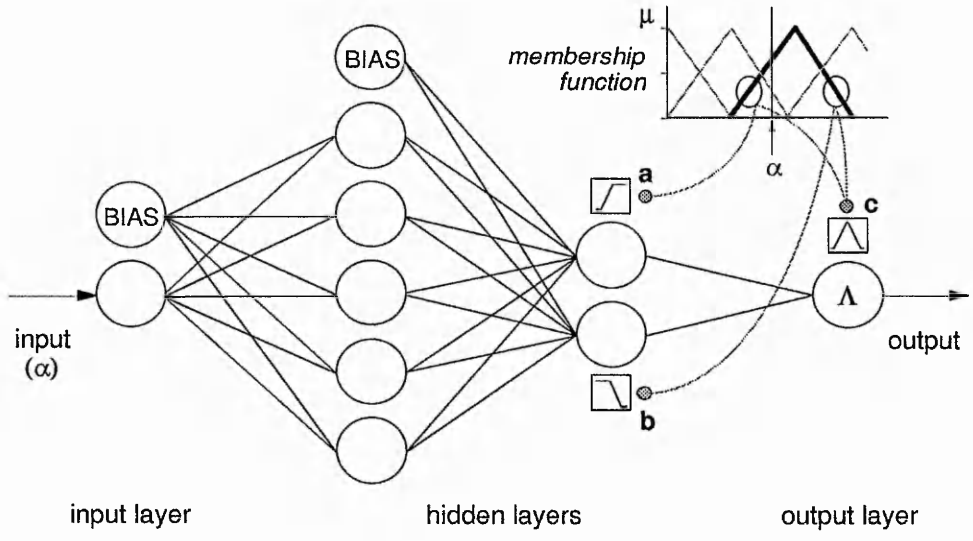


Figure 5-15: *Type A network architecture for generating fuzzy input membership function*

neurodes in the second layer. The supervised back-propagation training algorithm was engaged during the learning process. A primary element of the network for generating an input membership function is depicted in Figure 5-15.

From the neurodes in the second hidden layer of the network, *(a)* and *(b)* labelled in Figure 5-15, both sides of the membership function can be yielded, respectively. These two signals are then fed into a filter (labelled c) which simply does the *minimum* (Λ) operation. Finally, the entire membership function can be successfully generated and obtained from the output of the network.

(2) Type B

This type of structure employed is a multi-layer network with sigmoid function, eight neurodes in the first hidden layer and two in the second layer. The back-error

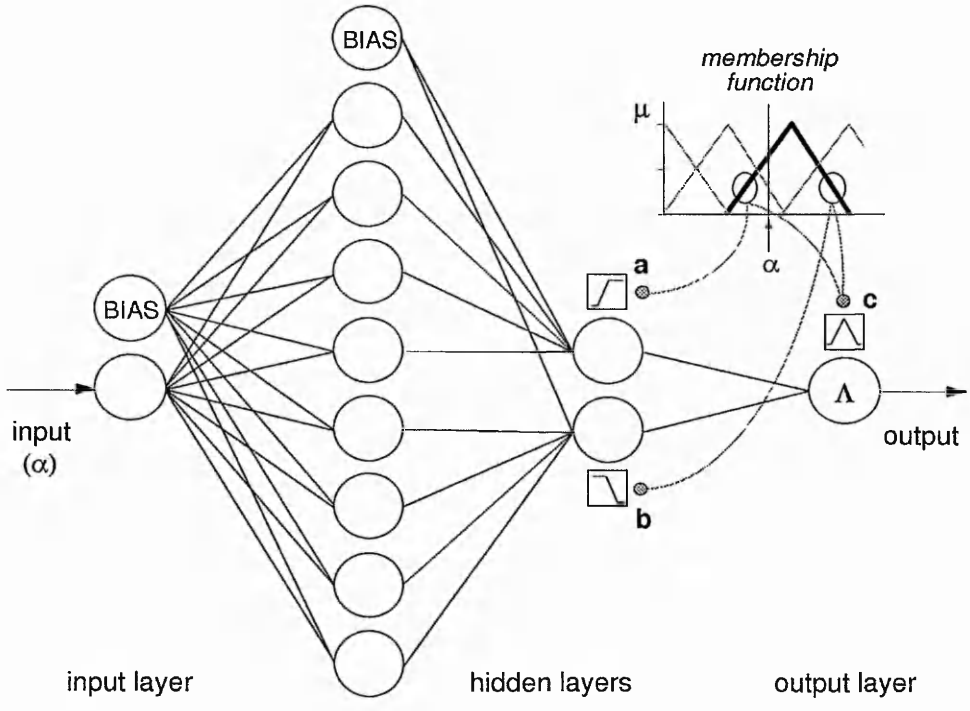


Figure 5-16: *Type B network architecture for generating fuzzy output membership function*

propagation learning algorithm is applied. Dissimilar to the Type A network architecture, partial connections between the neurodes in the first and second hidden layers are used. The left portion of the membership function can be obtained from the first neurode in the second hidden layer, *(a)* labelled in Figure 5-16, also the right portion of the function is generated from the second neurode *(b)*. The node *(c)* in Figure 5-16 combines these signals into a triangle membership function using a minimum operation.

The difference between these two types of structures is the matter of time taken for training and recalling the neural networks. It is obvious to find that the number of the neurodes in the hidden layers of the Type A architecture is less than in the Type B network. Accordingly, obtaining an output from the Type B network in the recalling mode (testing or running the trained network) will take a little bit longer than in the Type A structure. However, the time taken to successfully converge the Type A network is, according to the experimental results, more than five times longer than training the Type B network.

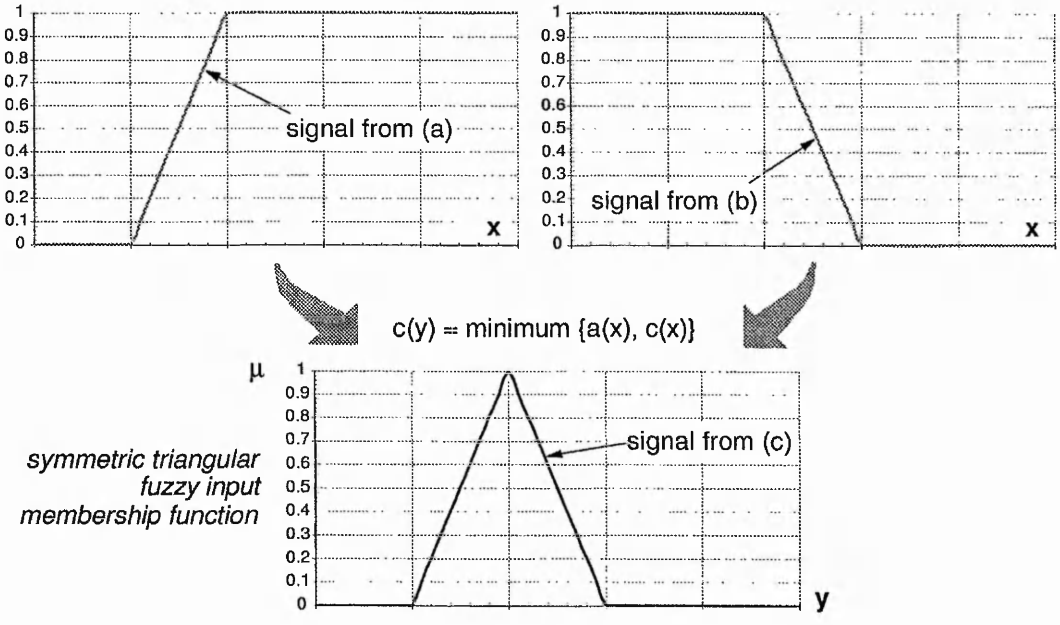


Figure 5-17: *A fuzzy input membership function generated by the neural fuzzy engine*

Depending on the requirements of the applications, if the neural fuzzy system is used for on-line training, the time spent for converging the neural networks becomes critical. Use of Type B network will be highly recommended there. On the other hand, if the size of the network structure or the time taken to recall (use) the trained network is limited, Type A network structure should be considered.

An example of using such a technique to generate the fuzzy input membership function (symmetric triangle) is illustrated in Figure 5-17. It is noted that by utilising the method not only the triangular or trapezoid input membership function can be created, but also the "bell" shaped membership function. Two example structures applying Type A and Type B network used six fuzzy variables (fuzzy sets) are illustrated in Figure 5-18 and 5-19, respectively.

5.3.4.2.2 Generating fuzzy output membership functions

A neural network architecture as depicted in Figure 5-20 is designed to generate a fuzzy output membership function such as trapezoid membership function. A three-layered fully

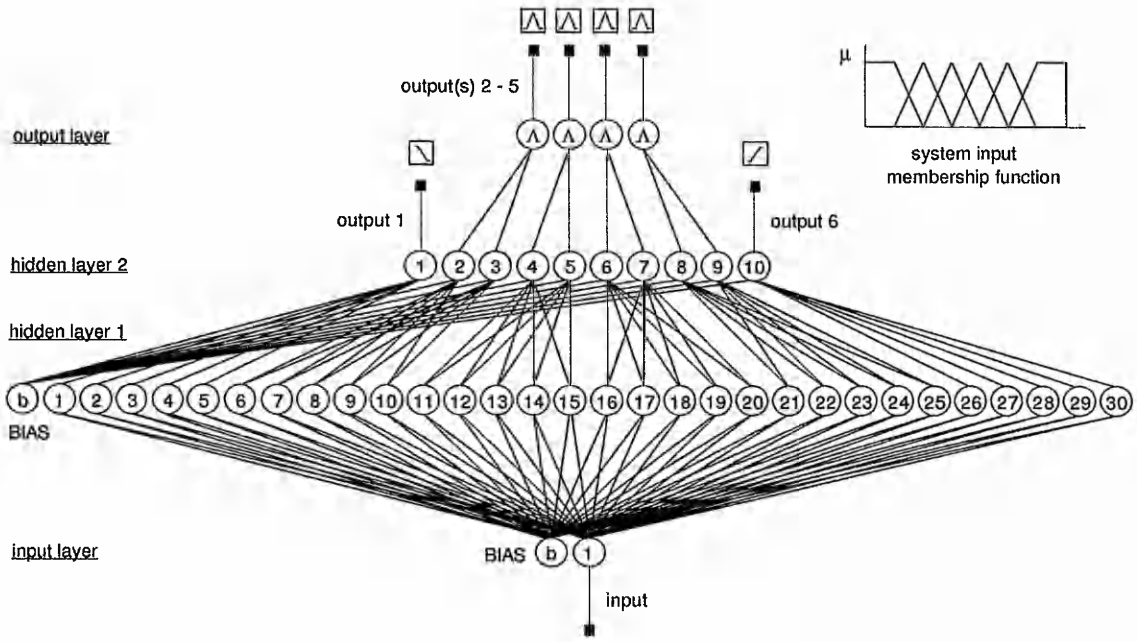


Figure 5-18: *Fuzzification process constructed by the Type A network*

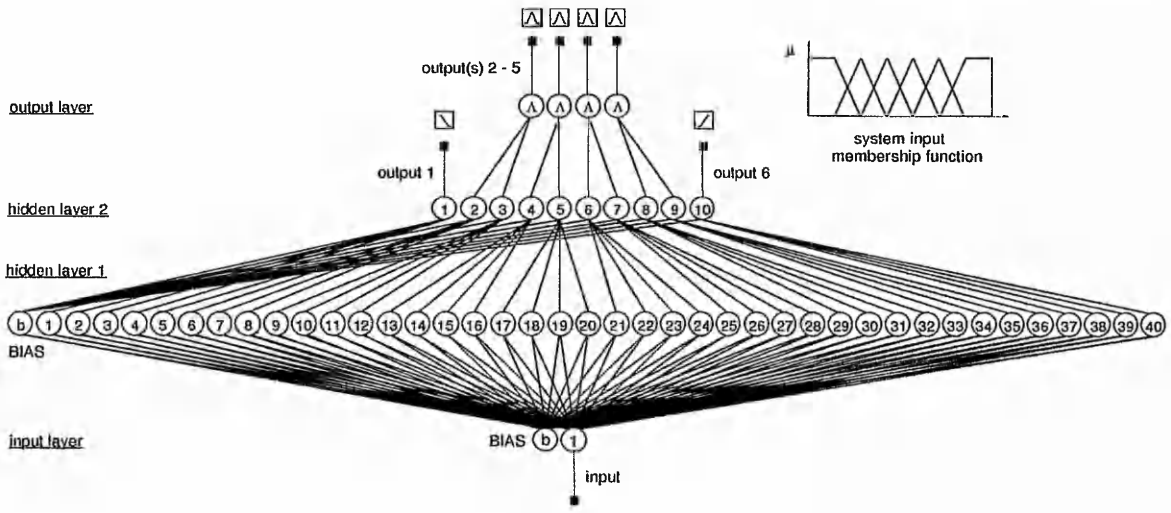


Figure 5-19: *Fuzzification process constructed by the Type B network*

connected back-propagation network including three neurodes in the hidden layer and two in the output layer with sigmoid function is used. Only a small number of representative data set is required for training the network - the scheme for producing the learning data sets will be described in Section 5.3.4.3.

Figure 5-21 shows the output data collected from the trained network. In the example a

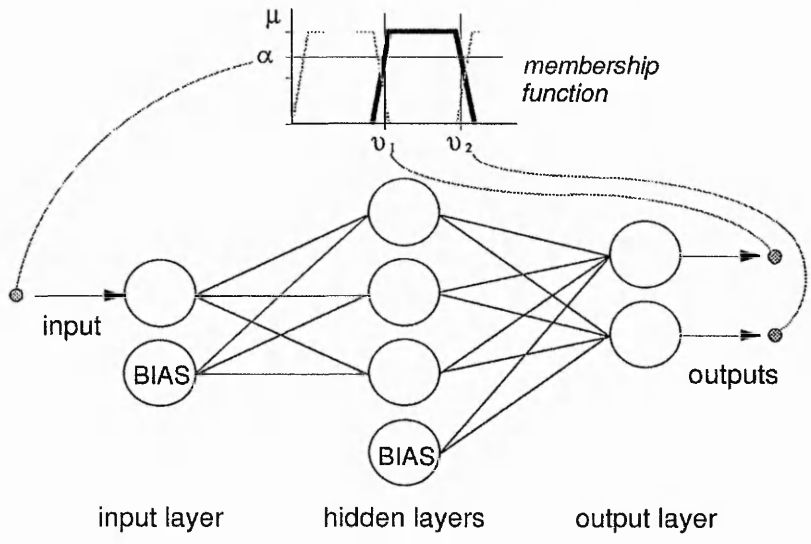


Figure 5-20: *Neural network architecture for generating a fuzzy output membership function*

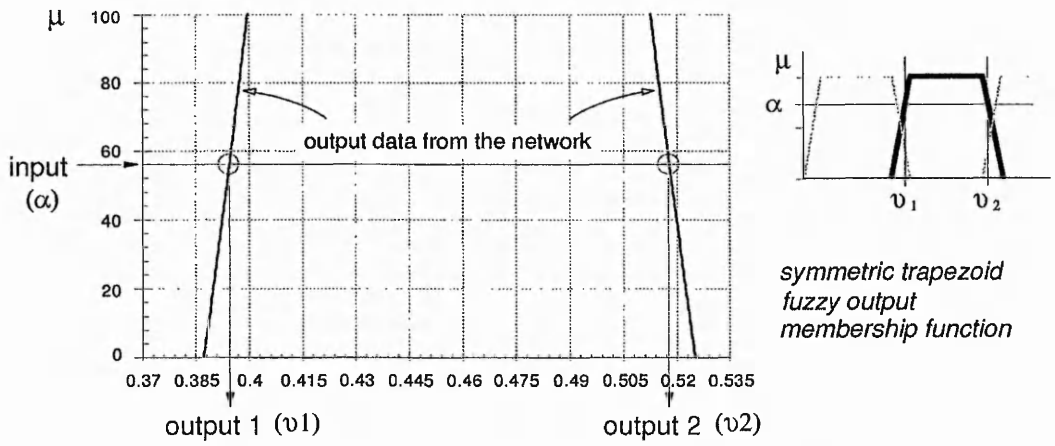


Figure 5-21: *A fuzzy output membership function*

symmetric trapezoid output membership function is utilised to generate the data set for training the network. It is very encouraging that only less than ten seconds is required to successfully converge the network (*delta* (δ) is set to 0.00000001) in which a 486 DX-2 PC running at 66 MHz is engaged.

Other shapes of fuzzy output membership functions, such as triangle and "bell" shaped function, are also investigated. The experimental results indicate that the network architecture employed here can define various shapes of the fuzzy output membership functions qualified in the project. An example of the network structure which produces a complete output membership function using six fuzzy variables is illustrated in Figure 5-22.

5.3.4.2.3 Connections of the rule base

An example of connecting the rule base of a neural fuzzy system which has two inputs and one output is illustrated in Figure 5-23. Nine rules can be derived from the FAM bank. As an illustration, the following two conditions can produce the same consequence:

$$\text{IF } input\ 1 = a \text{ AND } input\ 2 = A \text{ THEN } output = 1$$

$$\text{IF } input\ 1 = c \text{ AND } input\ 2 = B \text{ THEN } output = 1$$

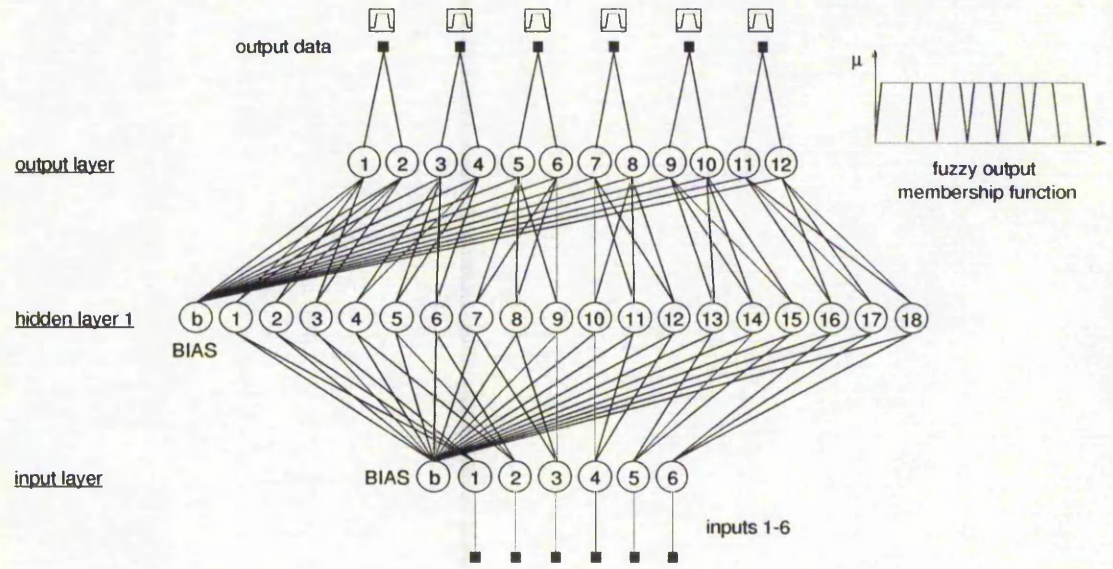


Figure 5-22: *Network structure for generating fuzzy output membership function*

A minimum (Λ) operation is used to merge neurode (a) and (A) (*aA* indicated in Figure 5-23). The same approach is applied to the neurode (c) and (B). The *output signal 1* can be acquired in terms of utilising the maximum (V) operation to join the node *aA* and *cB*. This procedure can be represented by the Equation 5-12. Using the same approach, the rest of the rules can be arranged within the network.

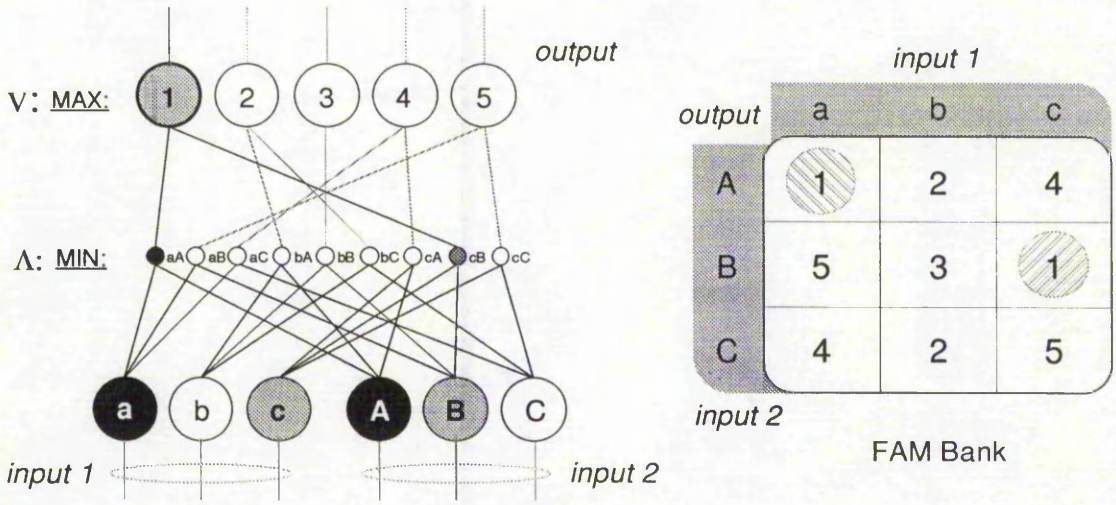


Figure 5-23: *Connections of the rule base*

$$Output\ 1 = \mathrm{V}\,[\,\Lambda(a,A)\,,\,\Lambda(c,B)\,] \tag{5-12}$$

5.3.4.3 Generating the training pattern pairs

In this section the scheme of generating a set of input/output pattern pairs for training the neural fuzzy engine is described. Symmetric triangles are used for the system input membership function. Also the system output membership function is arranged by the symmetric trapezoids. A program was written to automatically create the training data sets. The user only needs to specify some particular points of the membership function, the program can produce the actual learning data for the network. A normalisation process is included in this program.

As depicted in Figure 5-24, five points in the system input membership function and six points in the output membership function are selected (marked by circles). The program reads this data points and creates a set of learning data for training the system - twenty three

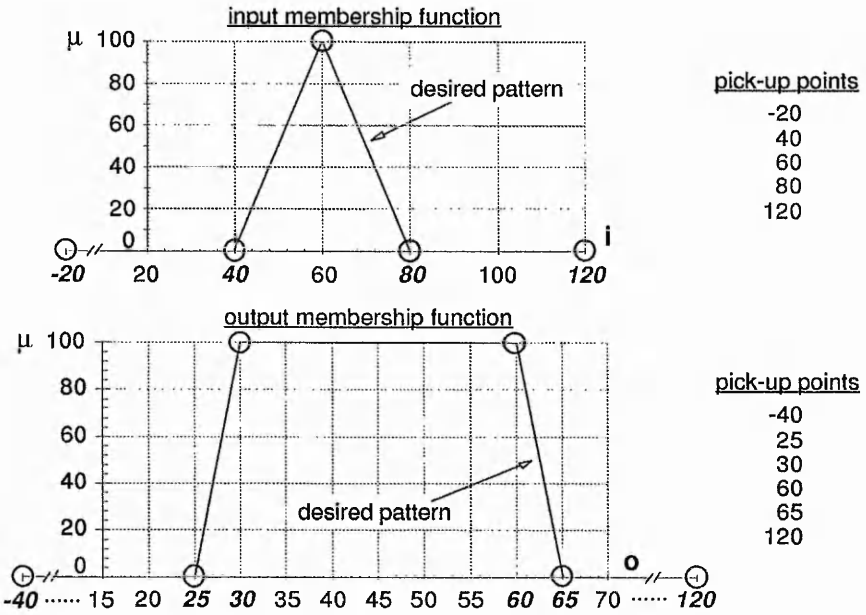


Figure 5-24: *The points have to be identified for generating the training data*

Input Membership Function						Output Membership Function					
Raw data			*Normalised data* ¤			Raw data			*Normalised data* §		
input	desired output		input	desired output		input	desired output		input	desired output	
-20	0	100	0.0	0.0	1.0	0	25	65	0.0	.4625	.5625
40	0	100	.42857	0.0	1.0	10	25.5	64.5	0.1	.46375	.56125
42	10	100	.44286	0.1	1.0	20	26	64	0.2	.465	.560
44	20	100	.45714	0.2	1.0	30	26.5	63.5	0.3	.46625	.55875
46	30	100	.47142	0.3	1.0	40	27	63	0.4	.4675	.5575
48	40	100	.48571	0.4	1.0	50	27.5	62.5	0.5	.46875	.55625
50	50	100	.50	0.5	1.0	60	28	62	0.6	.47	.555
52	60	100	.51429	0.6	1.0	70	28.5	61.5	0.7	.47125	.55375
54	70	100	.52857	0.7	1.0	80	29	61	0.8	.4725	.5525
56	80	100	.54286	0.8	1.0	90	29.5	60.5	0.9	.47375	.55125
58	90	100	.55714	0.9	1.0	100	30	60	1.0	.475	.550
60	100	100	.57143	1.0	1.0						
62	100	90	.58571	1.0	0.9						
64	100	80	.60	1.0	0.8						
66	100	70	.61429	1.0	0.7						
68	100	60	.62857	1.0	0.6						
70	100	50	.64286	1.0	0.5						
72	100	40	.65714	1.0	0.4						
74	100	30	.67143	1.0	0.3						
76	100	20	.68571	1.0	0.2						
78	100	10	.70	1.0	0.1						
80	100	0	.71429	1.0	0.0						
120	100	0	1.0	1.0	0.0						

¤ normalisation to values from 0.0 to 1.0
§ normalisation to values from 0.3 to 0.7

Table 5-4: *The actual training data sets for the neural fuzzy system*

data in the input membership function and eleven data in the output membership function are automatically selected. Table 5-4 lists the data pairs for training the neural fuzzy kernel to generate the membership functions illustrated in Figure 5-24. It is noted that the normalisation process for training the output membership function in the project is set to the values from within the interval [0.3, 0.7]. According to various experimental results, a shorter network converging time was found applying this value than using the conventional interval [0.0, 1.0].

5.3.4.4 The neural fuzzy engine

The architecture of the neural fuzzy engine which used two inputs and one output variables is described in detail in this section. For modeling the inference engine a multi-layer

feedforward network with sigmoidal elements was used. A back-propagation training algorithm was applied for training, with *learning rate* (η) of between 0.4 and 0.9 and *momentum factor* (α) of between 0.6 and 0.999. A random pattern presentation scheme was used for training. The connectivity matrices were initially randomised with values from within the interval [-0.3, 0.3].

Three fuzzy variables, such as left, middle and right, are applied to construct the input membership functions, also the output membership function are arranged by six fuzzy variables - NL, NM, NS, PS, PM, and PL. Figure 5-25 shows the system input and output membership functions together with the rule base. The synthesis of the neural fuzzy engine, presented here, goes through three phases:

1) Synthesis of neural network which realises the fuzzification of input variables;
2) Synthesis of neural network which realises the evaluation of the fuzzy control rules;
3) Synthesis of neural network which realises the defuzzification algorithm.

Using all the techniques described previously, the entire architecture of the neural fuzzy

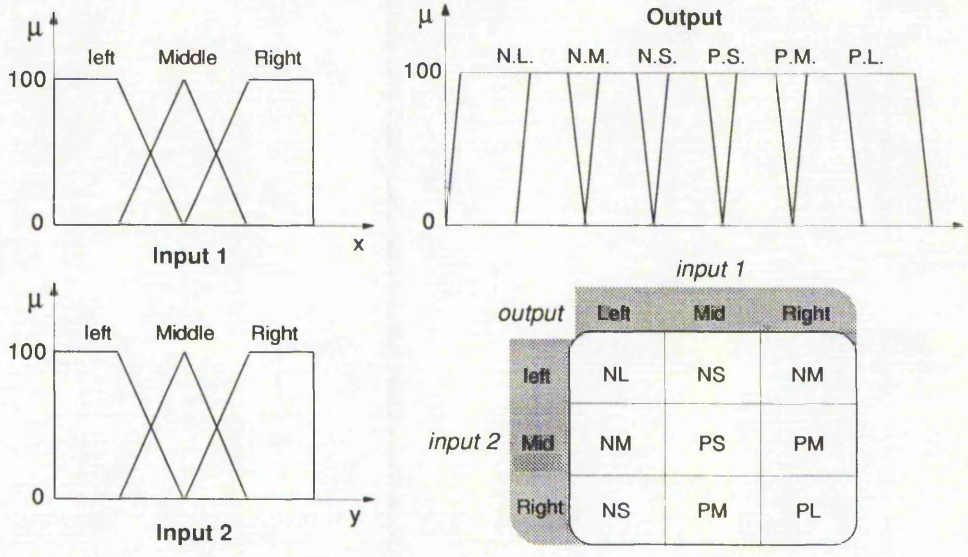


Figure 5-25: *Fuzzy membership functions and the rule base*

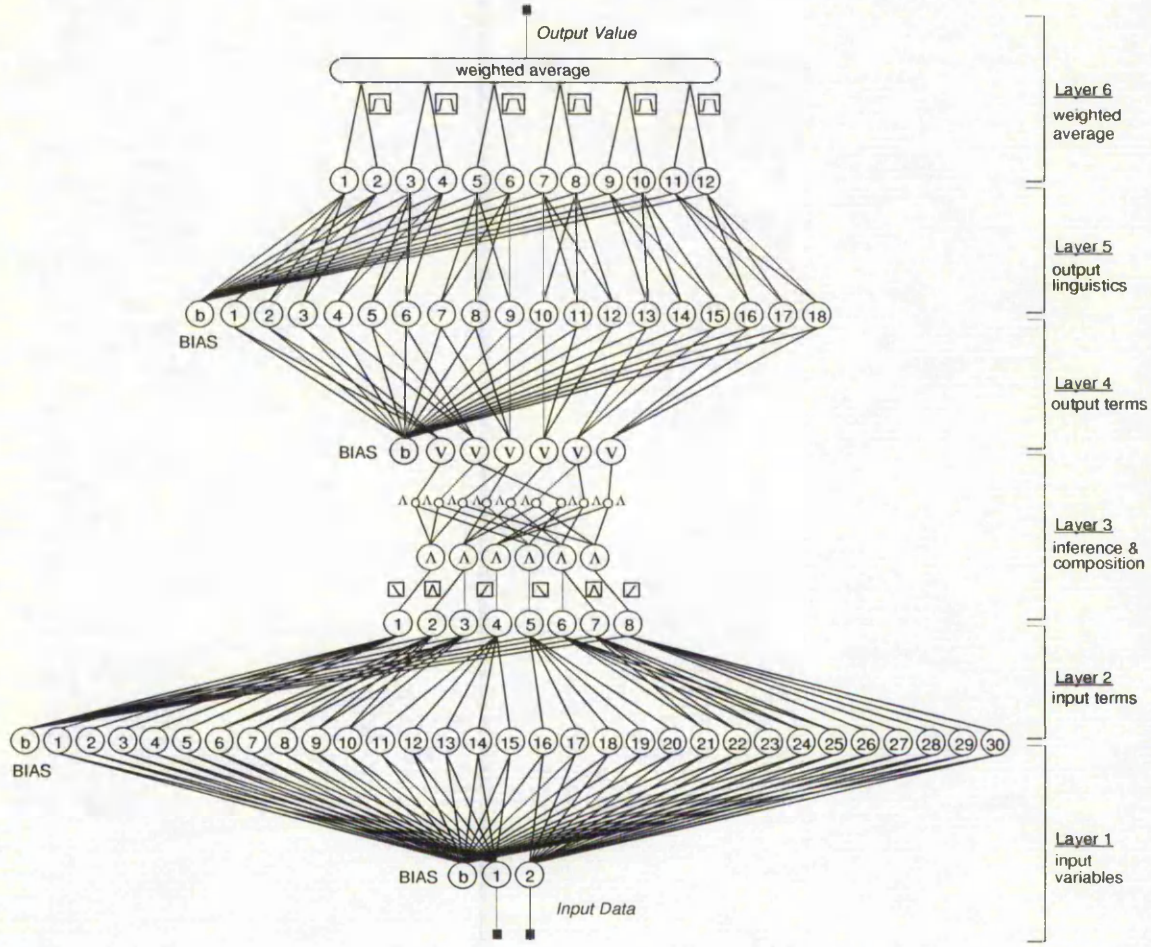


Figure 5-26: *Architecture of a two-input and one-output neural fuzzy engine*

engine as depicted in Figure 5-26 can be constructed. *Layer one* and *layer two* of the network read two external variables as well as transforming them into two fuzzified terms. The inference and composition process, *layer three* of the network, distributes the computed strengths of the signals from layer two and assigns to the fourth layer of the system. The defuzzification process then takes the output linguistics which is computed by the *fourth and fifth layers* of the engine, and calculates the weighted average. The crisp output (system response) of the engine can finally be obtained from the sixth layer of the network.

The author made the decision to implement the fuzzy inference system using neural networks for the following reasons:

1) Fewer data points are required because the network is able to generalise;
2) It allows off-line or even on-line training of the engine. This results in flexibility in membership function definition;
3) New data points may be added to re-train the network at a later time.

5.4 Configurations

The vision station constructed in the project consists of proprietary components such as two black and white CCD cameras, a four channel video multiplexer, an input/output plug-in card (a video frame grabber, 256K bytes of frame memory organised as 512x512x8 bits) and a video monitor. The above components are integrated within a desktop host environment. A CNC cutting mechanism (Pacer COMPACT 800+) is employed for driving the cutter (a *Spring Mounted Pen - SMP*, is engaged here). In addition an extra conveyor system is fitted on the machine for the transportation of strips of material (a lace or paper) under the vision system and the cutter. A M68000 based micro-controller controls both the cutting mechanism and the transportation system. The host system bus is used to provide the communication channel among the various elements of the system.

Figure 5-27 illustrates the configuration of the vision based machine control system developed. The host system receives the external video signal which is exchanged by the video multiplexer between two cameras, as well as displaying the captured image on the video monitor. The control data is then generated and transferred to the cutting mechanism

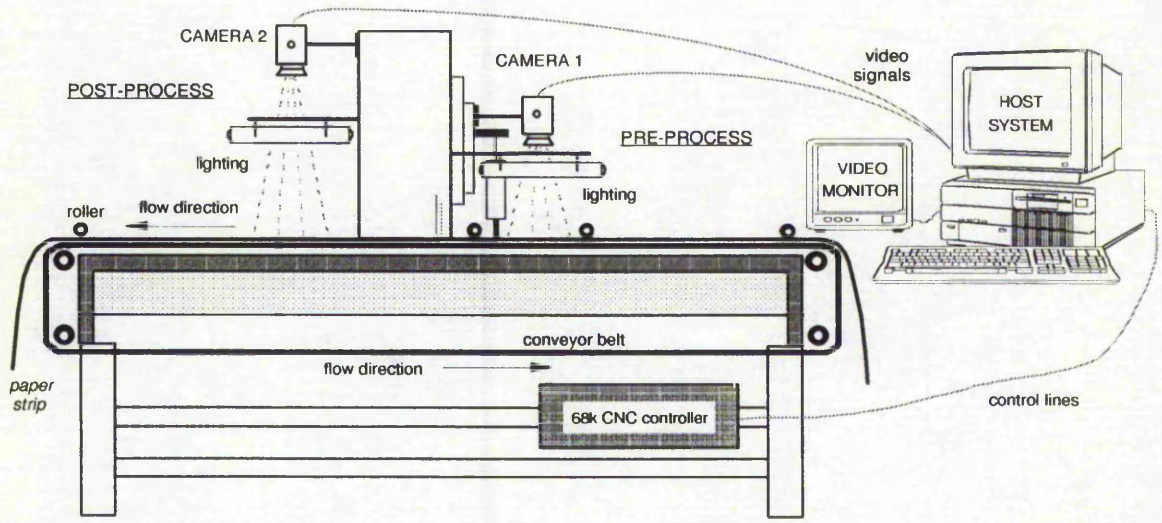


Figure 5-27: *Configuration of the tightly coupled vision based control system*

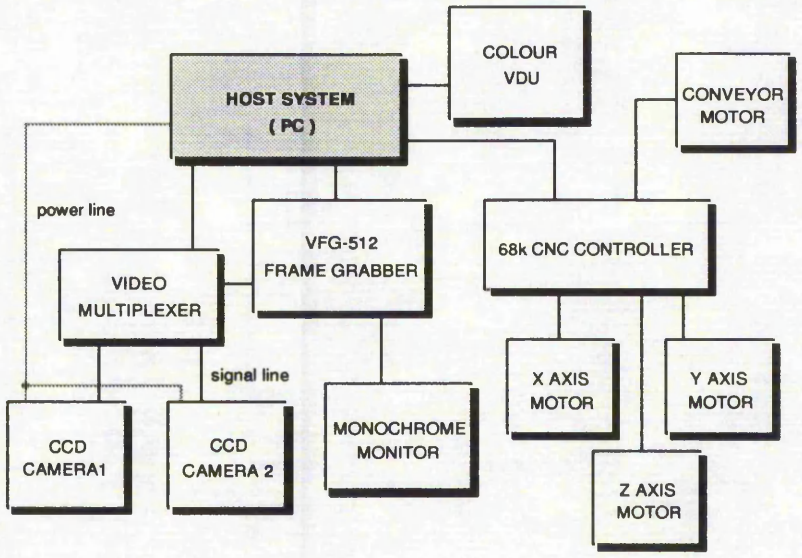


Figure 5-28: *Block diagram for the system overview*

and the transportation system (conveyor). The interconnection between these components is depicted in Figure 5-28 at a block diagram level.

The mode of motion control is closed loop. That is, the feedback signal is generated from the image input rather than any rotation sensors in the axes. Although the controller in the selected testing rig is not a real-time environment it provides a satisfactory means of developing a prototype which is, from the cost and performance point of view, effective for experimentation. The non-real-time nature of the controller of the testing rig results in compromising the overall system throughput and not being able to set the event priorities within the system to appropriate levels.

As mentioned previously a spring mounted pen is guided by the machine to draw a curve on the paper strip to emulate the distortion of the lace strip due to the cutting forces caused by a tactile cutter. As depicted in Figure 5-29, *camera one* picks up an object image and passes it to the pre-processing vision system which processes the raw image and generates the machine movement data. The cutting mechanism receives the data and drives the cutter following the obtained cutting path. The processed object is then transported under the

second CCD camera by the conveyor system. This camera captures an image of the processed object and transfers the data to the post-processing vision system in which the system examines the data and feeds back the analysed information to the pre-process vision system. The feed-back signal is provided to correct the subsequent cutting processes and results the precise handling of the object in correcting the motion.

5.5 Pre-processing Vision System

The pre-processing vision system captures a 256 grey scale image of the target path from *camera one* and temporarily stores it in the memory. An image bi-leveling (thresholding) operation is applied to transform the image into a black and white bitmap. The target path on the paper strip, as indicated in Figure 5-30, is then extracted by means of a line skeletonisation (skeleton) process (detailed description of these operations is presented in Section 2.4.1).

As a standard CCD camera mounted with a conventional lens is used in the project, the captured image tends to be distorted (refer to Section 2.5). The extracted cutting path is

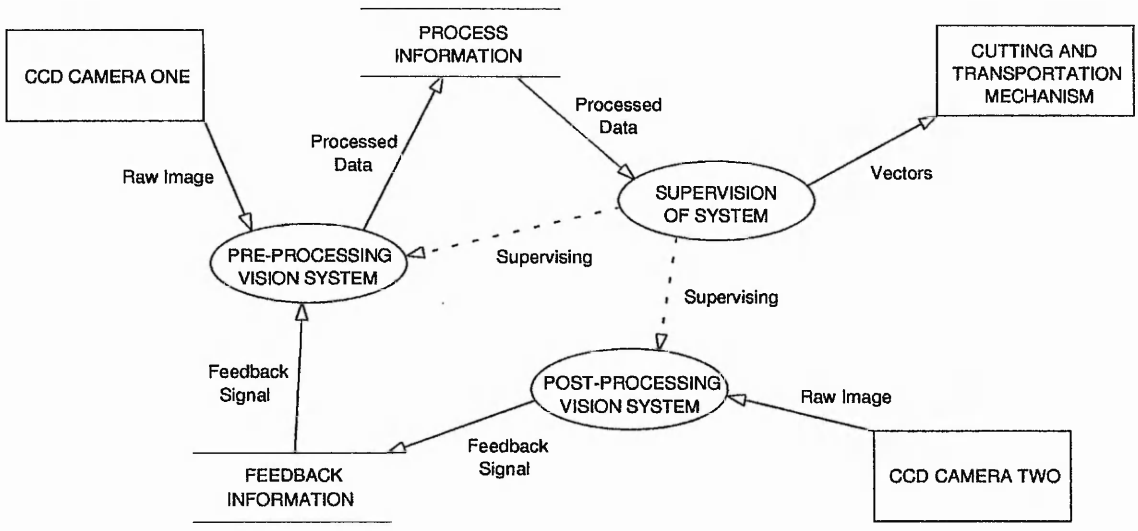


Figure 5-29: *Context diagram of the closely integrated vision based control system*

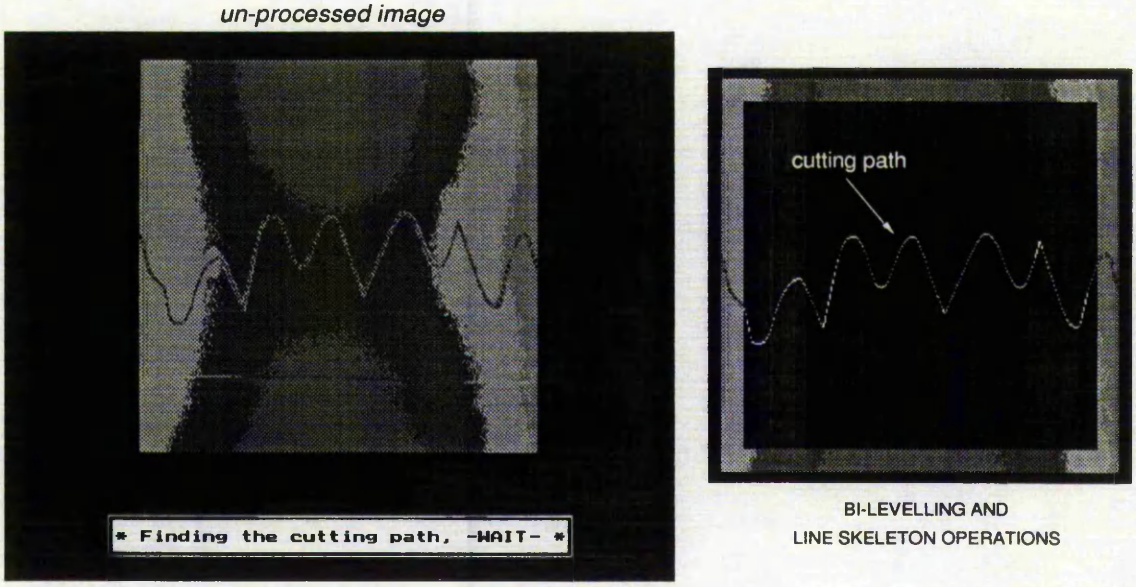


Figure 5-30: *Processes of the pre-processing vision system*

vectorised as well as corrected by a software filter, and transferred to the controller of the cutting mechanism. Figure 5-31 shows the path after correcting its distortion together with its vectorised drawing path for controlling the cutting mechanism. The machine moves at the *starting position* and pauses at the *end position* (indicated in Figure 5-31). When the cutter reaches the *capture point* within the path, in order to produce a new set of cutting data, an image of paper is captured and analysed. It should be noted that there is no feed-back information used here from the post-processing vision system to create the actual cutting path. The combination of the pre- and post-processing vision systems to produce the cutting path which is applied the feedback signal will be discussed later this chapter.

5.6 Post-processing Vision System

As the machine movement data is transferred to the cutting mechanism, the post-processing vision system starts to monitor the cutting (drawing) process. The analysed information is fed back to the machine console also. In order to emulate the distortion of the lace cutting process the spring mounted pen (SMP) is engaged to replace the tactile cutter.

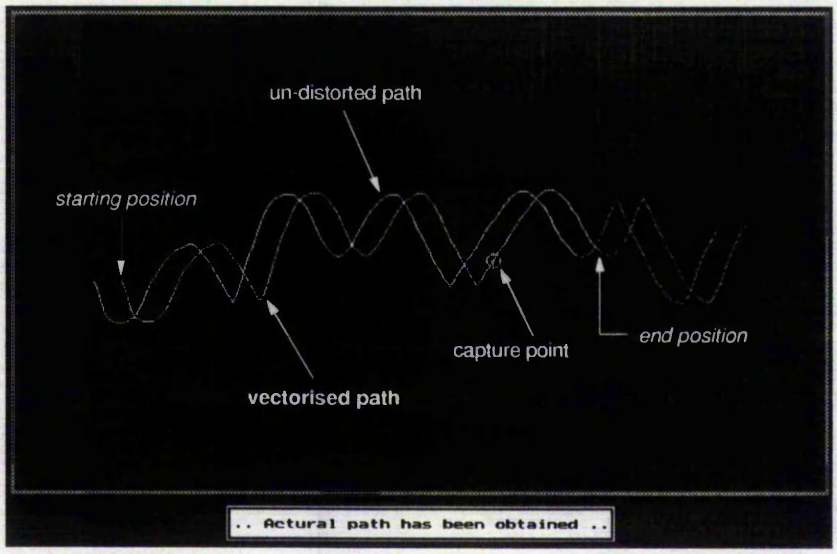


Figure 5-31: *The actual cutting path generated by the pre-processing vision system*

The distortion of the pattern is artificially generated by the spring. An experiment, where the SMP is guided to follow a square wave on paper, has been undertaken to evaluate the characteristics of the spring.

5.6.1 Square wave following

This experiment has been carried out to investigate the characteristic of the SMP which artificially creates the distortion of the cutting process for emulating the material stretch. The SMP is attached on the Z axis of the CNC machine controlled by a PC console. The SMP is led to follow a pre-defined square wave.

When the cutting mechanism is preset, the machine console controls the SMP to draw a path (as illustrated in Figure 5-32) on paper. Obviously, due to the flexibility of the spring, the *path-following-errors* appear between the desired square wave and actual drawing line (see Sample One, Figure 5-32). In addition, Sample Two in Figure 5-32 illustrates two drawn paths taken when the Z axis of the CNC machine is set to different contact pressure.

The reset of the cutting head can cause the SMP to bear different levels of pressure hence produce different drawing patterns.

It can be observed that the path-following-error will be generated when the direction of the drawing is changed - the larger the angular variation of the path following, the larger is the error. The amount (magnitude) of the path-following-error generated is depended on the characteristic of the spring engaged, the pressure on the SMP, and the frictional force in between the tip of the pen and the paper (refer to Figure 5-33). As any one of the system coefficients is altered, the result of the SMP drawing will be entirely different.

5.6.2 Information feedback

The post-processing vision system is engaged here to capture as well as examining the image of the resultant path and the target pattern. The deviation between the paths (the path-following-errors) is fed back to the host system. The host PC analyses the information

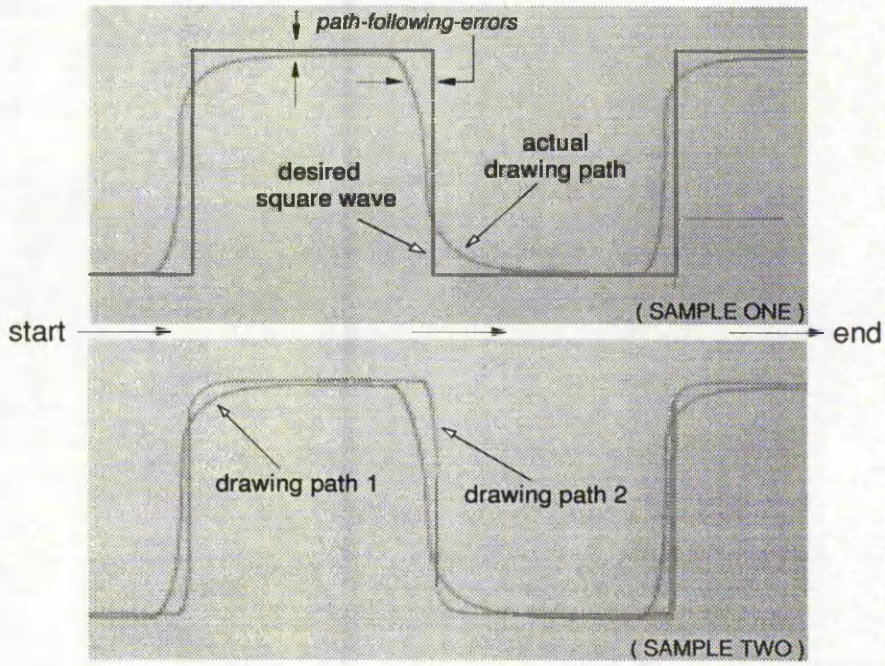


Figure 5-32: *Samples of square wave following process*

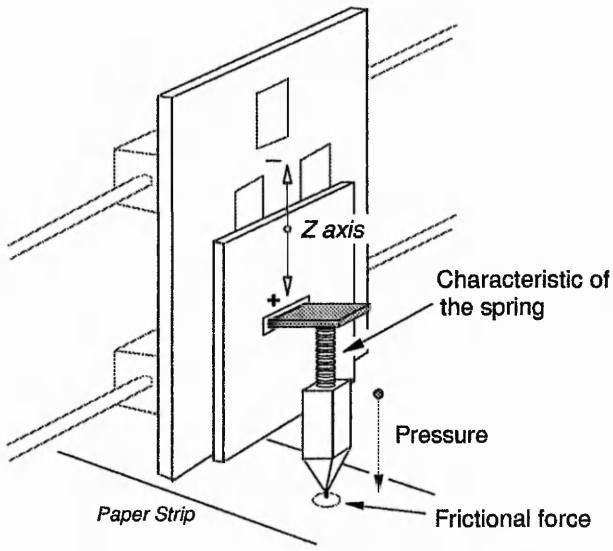


Figure 5-33: *Three system coefficients affects the path following process*

obtained from the post-processing vision system and determines a necessary compensation to improve the operation for the subsequent process (closed loop control). This analysed (learned) data is then used to build a record that can be utilised for further analysis.

While the resultant path is transported beneath the camera of the post-processing vision system, an image of the paper strip is captured and stored in the memory. The path-following-error appearing in this frame is detected by a software analyser developed by the researcher (the program will be described in Section 5.7.3.1). Since the extreme complexity of the path-following-error caused by the SMP is affected by three system parameters, such as *characteristic of spring*, *the pressure*, and *the friction force* (Figure 5-33), it seems to be very difficult to use conventional methods, where physical sensors are applied to measure all of the system coefficients hence find the related information to correct this error. In order to overcome the problems of complexity, the inexact algorithms, namely fuzzy logic, neural fuzzy technique, and neural networks, are considered.

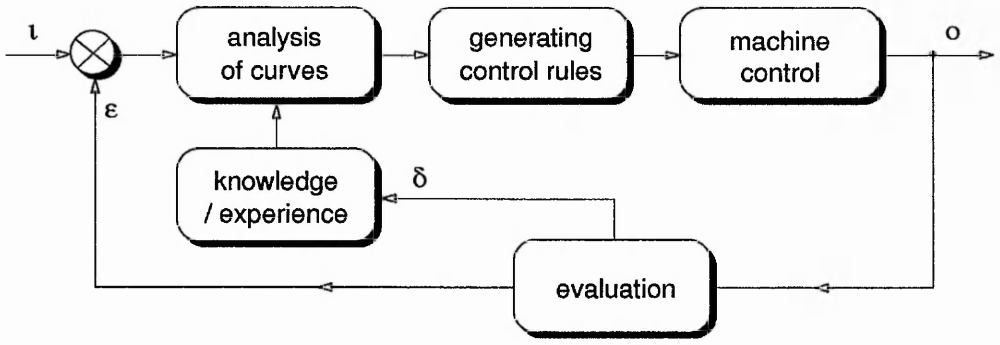


Figure 5-34: *The control scheme of a close-loop learning process*

5.7 Closely Integrating the Remote Sensing Based Control

As stated previously, two cameras together with the CNC machine are integrated using a host computer. The host system supervises the data flows and control actions among all the elements. The tightly integrated system is divided into three sub-systems: pre-processing vision system, post-processing vision system, and motion control system. In this section, the author describes the techniques derived for incorporating two vision systems to correct the path-following-errors.

5.7.1 Introduction

The principle aim is to design a vision based control system, in the most natural way, in terms of imitating the operator's control action and experience or knowledge. Imagining that the host system controls the cutting mechanism fitted with a SMP to draw a line on paper with no correcting action. Path-following-errors occur between the desired path and the actual drawing line. Utilising the previous experiences, the human operator analyses the difference between the paths and determines a possible correcting action. The operator then controls a joy-stick (X/Y axes) of the machine tracing the desired path as close as possible. This approach, as depicted in Figure 5-34, can be described as follows:

1) Analysing the difference between two curves, the path-following-error is represented as linguistic description, e.g. *the error between curves on X coordinate is huge*;
2) Expressing the operator's control actions by a set of control rules;
3) Control the machine guiding the SMP to following the desired path;
4) Repeat stages 1) to 3) to get the drawn curve closer to the desired path (learning procedure).

A human operator usually controls a machine based on his experience and/or knowledge which normally can be expressed as a set of control actions (rules). By modeling an operator's control actions to design a computer based control system, one does not need to understand how the system parameters (i.e. pressure, frictional force, etc.) physically affect the performance of the controlled system in detail. A driver, for example, who does not need to understand the frictional force between the tires of the car and the road, or the weights of the passengers, can drive the vehicle quite well. Applying this idea to implement the system, the author proposed two different schemes attempting to solve the SMP problem:

1) Correcting the error based on analysing the shape of the curve;
2) or based on analysing the segments of the curve.

Figure 5-35 shows a resultant path extracted from a paper image. Different shapes of *curves* (portions of the path, marked by squares in Figure 5-35) is detected. The first proposed approach is to recognise / understand the shapes of the curves scanned (captured by the camera of the pre-processing system), and the difference between the desired path and the actual drawn line. When the system detects a curve similar to the one already in the record, the linked path-following-error of the record can be used to calculate an appropriate correcting action for the new curve. The detailed description of the method is discussed in the next section.

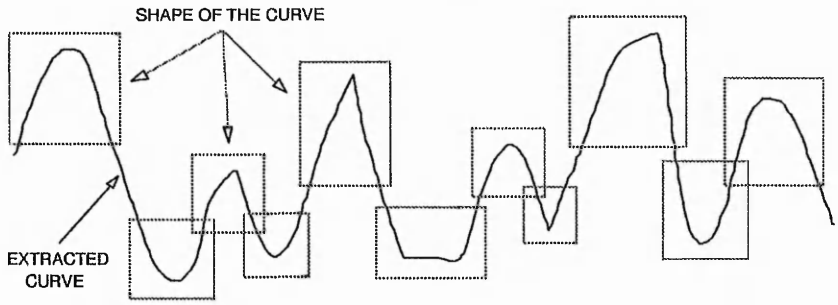


Figure 5-35: *Various shapes of curve from a scanned path*

The second proposed method is based on analysing the segments of the curve. Since the resolution of the camera in the pre-processing vision system *(Pre-PVS)* is only 256 by 256 pixels, the maximum points to represent the scanned curve is limited by this factor. An example of the vectorised curve pattern is illustrated in Figure 5-36 (refer to Chapter Two for the detail). This method analyses only a segment of the path at a time rather than the entire curve. A segment contains two or three data points (pixels) in the scanned path. By examining the locations of these points relative to the each other, the system can determine a possible modification to correct the path-following-errors. Based on this concept, two different schemes were designed and implemented. The first approach suggested is based on manipulating three consecutive data points (pixels) on the path. By calculating the angle between these points, the correcting path can be predicted. The second method investigated is based on the analysis of two adjacent data points (a segment of a path). Connecting a line

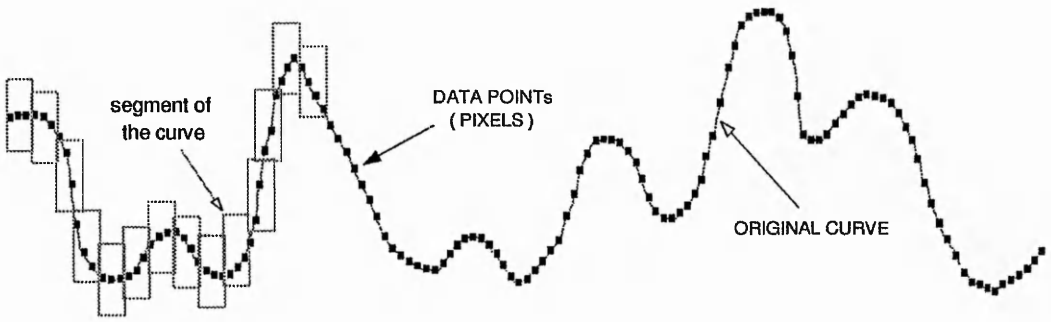


Figure 5-36: *A vectorised curve pattern*

between these two points, the angles between the line and the X / Y coordinates are used to determine a necessary correction for this segment.

As the Pre-PVS (Pre-Processing Vision System) grabs an image of the paper strip, the *curve extractor* (refer to Section 2.4.1) analyses the 256 by 256 gray scale image and picks up the drawing path. This path information is vectorised by the *path generator* and transferred to the 68k controller. The SMP is conducted by the two and half axes cutting mechanism which drives the SMP to draw a curve overlapping he desired path. When the processed curve passes under the camera in the post-processing vision system, a frame of 512 by 512 gray scale image is taken and saved. It is worth noting that different resolutions of the cameras are chosen to design the pre- and post-processing vision systems. The reason for using higher resolution in Post-PVS than in Pre-PVS is that much detailed information is required in the Post-PVS for analysing the difference between the two paths.

5.7.2 Using curve shape analysis

As the curve on paper is scanned by the Pre-PVS, 226 data points (refer to Section 2.5.3) are used to represent and record the path. The *top* and *bottom points* of the path are detected which are used to calculate the *centre points* (Figure 5-37). A path segment between two centre points is defined as a shape of curve. This *data reduction operation* was developed and employed to reduce the number of the vectors in the curves. Seven points (vectors) are used to record a curve of the path. After separating all the curves within the scanned path (as shown in Figure 5-37 (e), ten different shapes of curves are found), a *vector direction encoding technique* is applied to encode the curves into sets of numbers. Thirty six directions encoding is used in the experiments. The coded curves together with their path-following-errors are linked and recorded.

The idea of this method is to utilise the record of the processed curves as well as their linked path-following-errors. When a shape of a new detected curve is matched with a curve

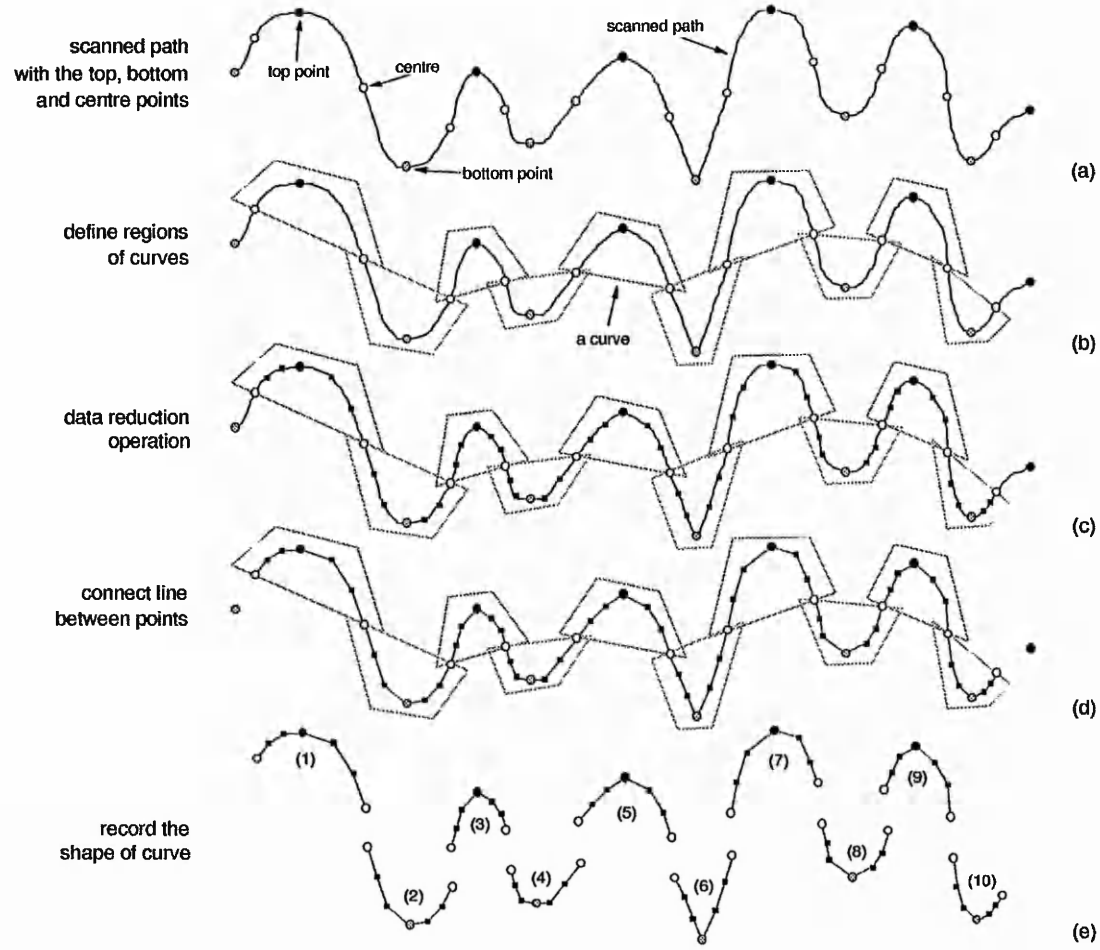


Figure 5-37: *Example of detecting the curve shapes from a scanned path*

in the record, its linked error data is provided to be a reference for calculating the correction. For example, a new scanned curve is similar to the registered curve A in the record, and its linked error record A is -5 pixels. Consequently the system can predict a correcting curve by adding extra five pixels to the scanned curve in the desired path.

The problem of using such approach is that, according to the experimental results, it is very rare to find that two curves can be completely matched. However, a alternative method is to employ an approximately matching scheme instead of using the exact mapping technique. When a new curve is found similar (say 90 per-cent matching) to a registered curve, the linked path-following-error is used to correct the error between lines. Unfortunately, the results of applying this scheme are unsatisfactory. Furthermore, since the

learning process (for correcting the errors) is heavily dependent on the shapes of the curves scanned, it is very difficult to judge when the system is 100 per-cent trained - the error may appear when the system detects a new shape of curve. This results that the system becomes unreliable and ineffective. Assuming we can record all different shapes of curves and their corresponding errors, a huge database has to be created for registering the information. As a result the system needs to take a long time to search the mapped patterns throughout the record. This seems not very practical for use in a real-time machine control system. Due to unsatisfactory results obtained from this scheme this approach was abandoned.

5.7.3 Using path segment analysis

Two schemes based on examining each individual segment of the path have been designed and implemented. The first method is based on manipulating the angles between *three consecutive coordinates* in the path *(3VMethod)*. The second method is based on the analysis of the angles between the *two adjacent coordinates* and *the X/Y coordinates (2VMethod)*. The overall system diagram is presented in Figure 5-38.

The path-following-error is firstly detected by a *colour-curve extractor* described in the next section. The error data is fed into the A.I. Engine One which analyses the difference between the paths also decides the *amplitude of the correction* for further process. Camera One, as indicated in Figure 5-38, in the Pre-PVS is triggered to capture a new frame of the desired path on paper. At this time, before the extracted curve is sent to the *path generator*, the segments of the extracted path are passed to the A.I. Engine Two which determines the pattern (shape) of the correction for the path. Both the amplitude and the pattern are used to generate a *predicted correcting path.* Finally, the path generator vectorises the predicted path and produce the movement data to the machine.

According to the information provided by the A.I. Engines, the CNC machine drives the SMP to following the desired path. A small path-following-errors may still appear in the first

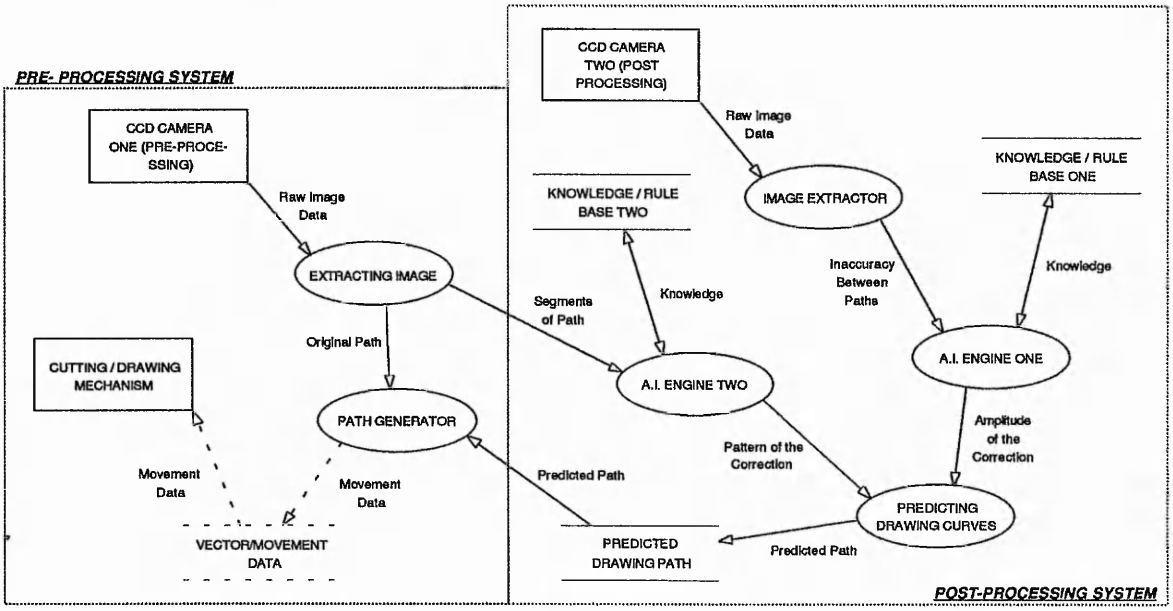


Figure 5-38: *Incorporation of vision and motion control systems*

corrected frame. Continually applying this approach the error information is repeatedly supplied to the A.I. Engines to calculate more accurate correcting actions (learning process) until two paths are finally matched.

5.7.3.1 Colour curves differentiation

Two different colour felt-tip pens are utilised in the experiments to draw lines on paper. The black pen is used to draw a desired path on a white paper strip by the operator. The spring mounted pen (used yellow or red) is driven by the CNC machine to draw a second line on paper. The Post-PVS is trigged to capture an image of these two paths and stores it in memory. Since the vision system, engaged in the project, is only a monochrome frame grabber, it is not an easy task to distinguish two different colour lines (e.g. black and yellow) in terms of analysing the gray scale image. A *colour curve differentiating technique*, therefore, has been developed to solve this problem.

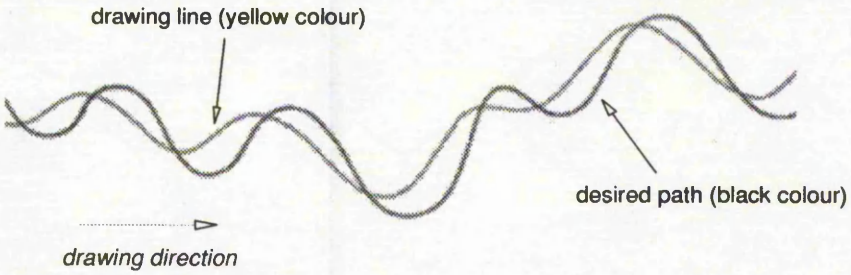


Figure 5-39: *Example of Black and yellow colour lines grabbed by a monochrome frame grabber*

As illustrated in Figure 5-39, a black line and a yellow line on paper is captured by the frame grabber and saved as a 256 gray scale image. By visual inspecting it is clear to see that the gray scales of these two lines are very similar (comparing it with the original colour picture). Figure 5-40 shows the gray levels representing the black and yellow lines. Two areas of overlapped gray levels can be found in this instance. The overlapped sections cause the difficulty in extracting the patterns. The method developed to differentiate these two colour lines is as follows:

1) Selecting the thresholding from point (a) to point (d) (indicated in Figure 5-40), the gray-scaled image can be converted into a black and white bitmap by using the

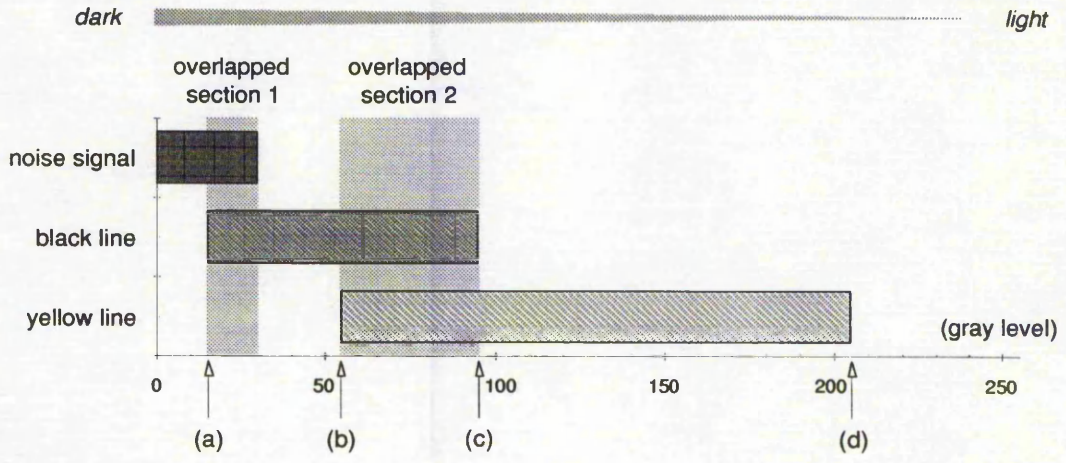


Figure 5-40: *Example of different gray levels represent colours*

average intensity analysis mentioned in Section 2.3.2. After removing the 'noise' in the bitmap, line skeleton operation is applied to process the pattern (Figure 5-41 (a)). The location of the processed pattern (two extracted lines) is recorded.

2) Setting the thresholding between point (a) and (c), the strip bi-leveling operation is, again, used to transform the image (Figure 5-39) into a bitmap. After noise

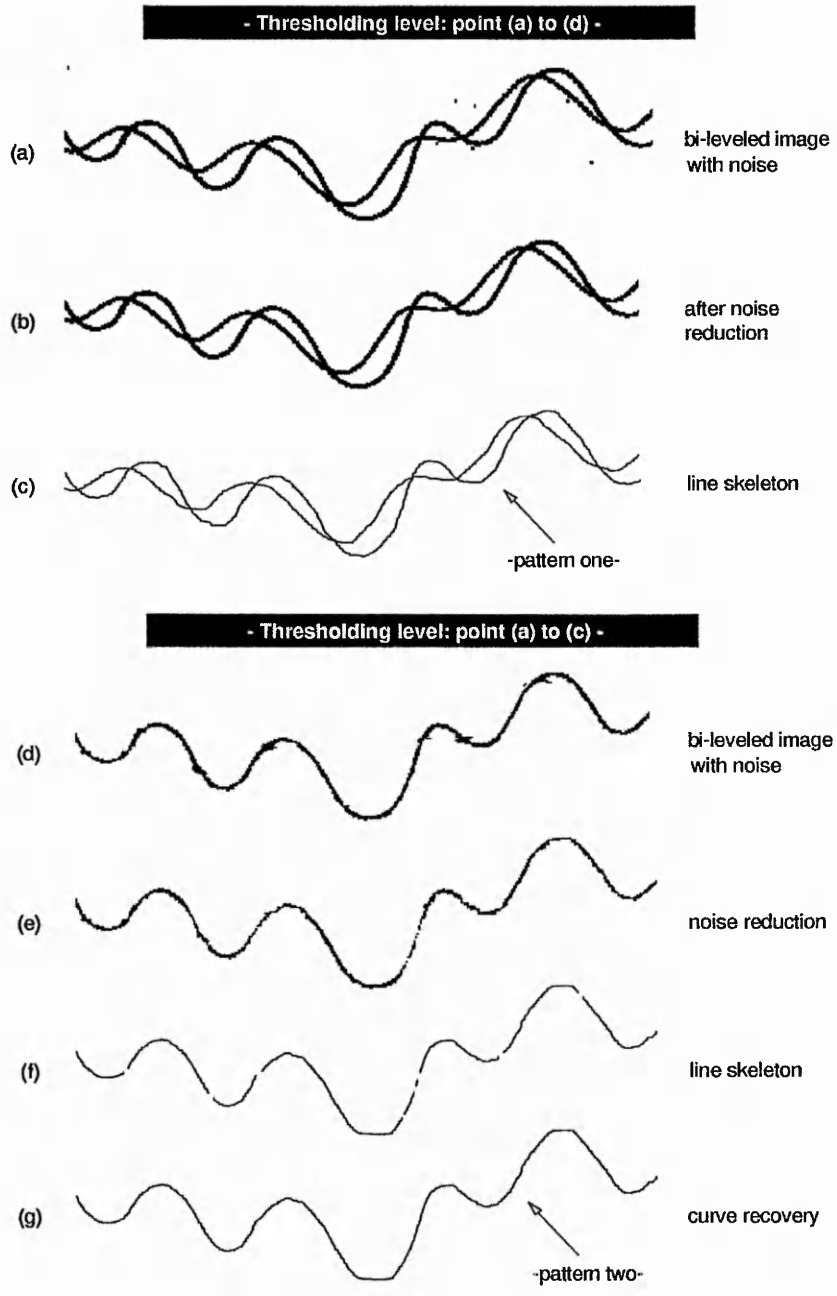


Figure 5-41: *Example of distinguishing two colour lines, (a)*

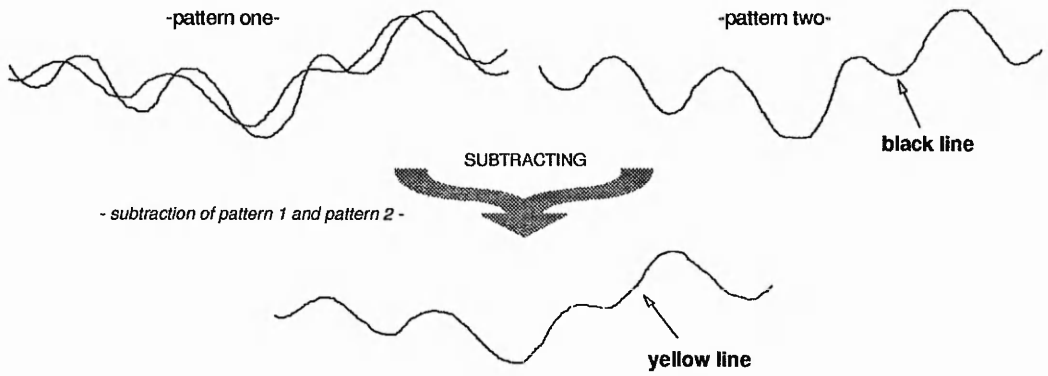


Figure 5-41: *Example of distinguishing two colour lines, (b)*

reduction process and line skeleton operation, the processed pattern may be split into several segments (see Figure 5-41 (a), step (f)). Connecting arcs between two adjacent segments, the *black line* can be reconstructed and stored in the memory (curve recovery process).

3) Subtracting the first and the second patterns, the *yellow line* can be obtained (Figure 5-41 (b)).

5.7.3.2 Three-vector method

This method is based on analysing the angles between three consecutive coordinates of the scanned path. As already mentioned, the scanned path from the Pre-PVS is digitised and stored in the memory. Two hundred and twenty six coordinates (vectors) are used to represent the scanned path (desired drawing path). As the angle between these coordinates is small, the correction added on the desired path by the operator is large. Yet if the angle is large (near 180 degree), the correction is small. We can derive a set of control rules from human operator's control actions. This is the key to solve the SMP path following problems.

Figure 5-42 shows the correlation between the angle and the correcting magnitude. As we can see, Angle A has the smallest angle (approximately 70 degrees) and Angle D has the biggest angle (230 degrees) in the example. When the angle is equal to 180 degree (Angle

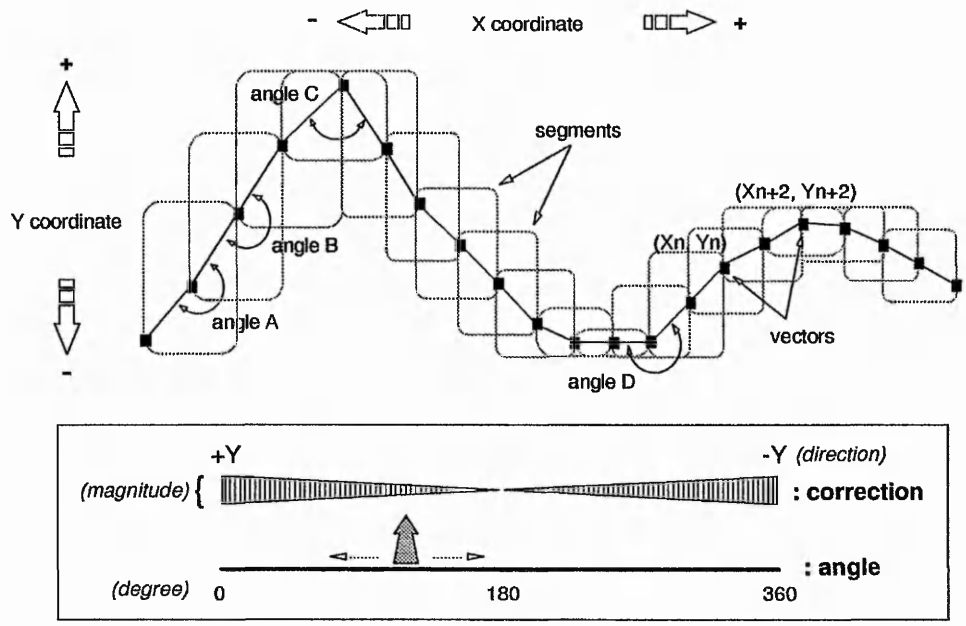


Figure 5-42: *Correlation between angle and the correcting energy (direction and magnitude)*

B), the correcting magnitude is set to zero. If the angle is close to 0 (or 360) degree, a large positive (or negative) correcting magnitude will be added to the Y coordinate of the scanned path. Equation 5-13 and 5-14 summarise the above.

$$Predicted\ Path = \sum_{i=1}^{No.\ of\ Segments} Segment(i)_{predicted\ path} \tag{5-13}$$

$$Segment(i)_{predicted\ path} = Segment(i)_{original\ path} + \sum_{n=1}^{i} (correcting\ energy)_n \tag{5-14}$$

where i is the i^{th} segment of the path. From the equations above it is clear that each segment of the predicted path is affected by its neighboring coordinates. In fact, if one of the correcting segments is wrong, it will affect all the rest of the corrections and produce unsatisfactory results.

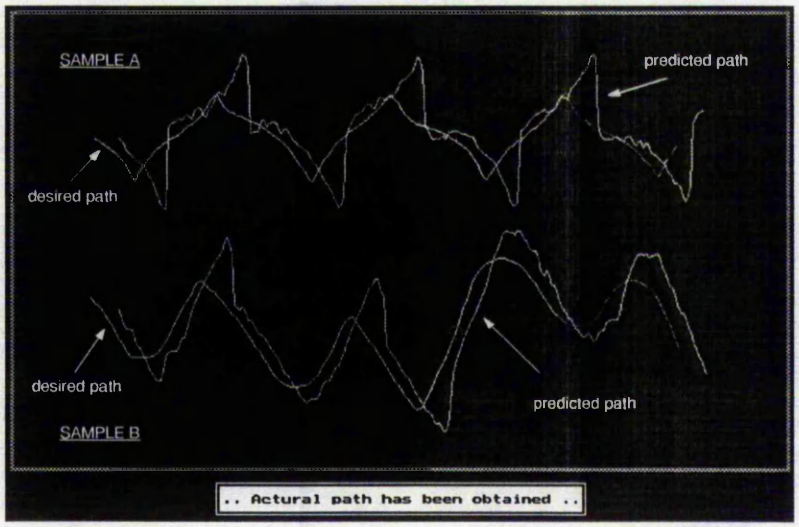


Figure 5-43: *Example of predicting a corrected drawing path using 3VMethod*

Figure 5-43 illustrates two sample paths together with their predicted patterns. Sample A is a regular shape of curve extracted from a lace pattern. Sample B is an irregular path which can be a triangle, trapezoid, arc, square, straight line, or a mix of all the shapes. It is believed that starting from analysing the results of tracking regular shapes of curves can obtain better and unmistakable experiences than directly handling irregular paths.

In the experiments, various control parameters were created and used in the testing. By adjusting these parameters, the magnitude of the correction relating to the angle of the coordinates is affected. Changing these system parameters will directly affect the performance of the SMP path correcting algorithm. As the predicted path data is transferred to the machine controller, the SMP is driven to draw a second curve line (as depicted in Figures 5-44 to 5-46, drawing from left to right).

Compared to the pattern drawn without the correction, it is obvious that about 60 to 80 percent of the path-following-error has been successfully removed in terms of applying the 3VMethod. Indeed, small sections of curves completely match. However, we might be asked why use of this method cannot produce completely satisfactory results, or why it

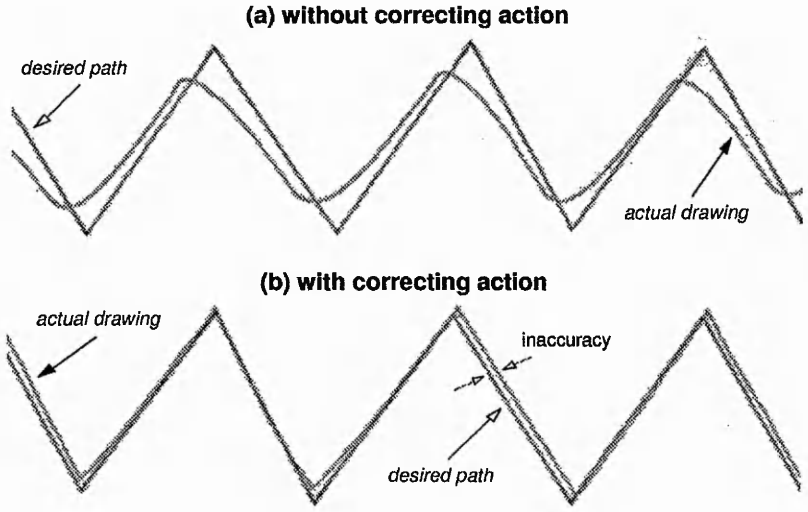


Figure 5-44: *Example A, using 3VMethod to correct the error*

cannot remove the rest of the error in the paths. From Equation 5-14 we understand that by applying this algorithm each predicted segments is affected by others. This means when a segment of the predicted path is inaccurate, the rest of the segments are affected and yield inaccurate outcome. A clear example of the situation is represented in Figure 5-46.

In addition, this method only adds the corrections in the Y coordinate of the scanned path. If the angle of the path is changed dramatically (degree of the angle is very small), no matter how big a correcting magnitude is added to the Y coordinate, the path-following-

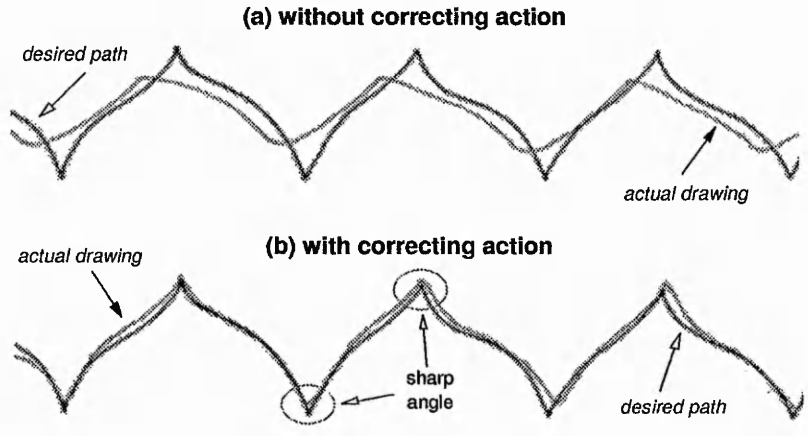


Figure 5-45: *Example B, using 3VMethod to correct the error*

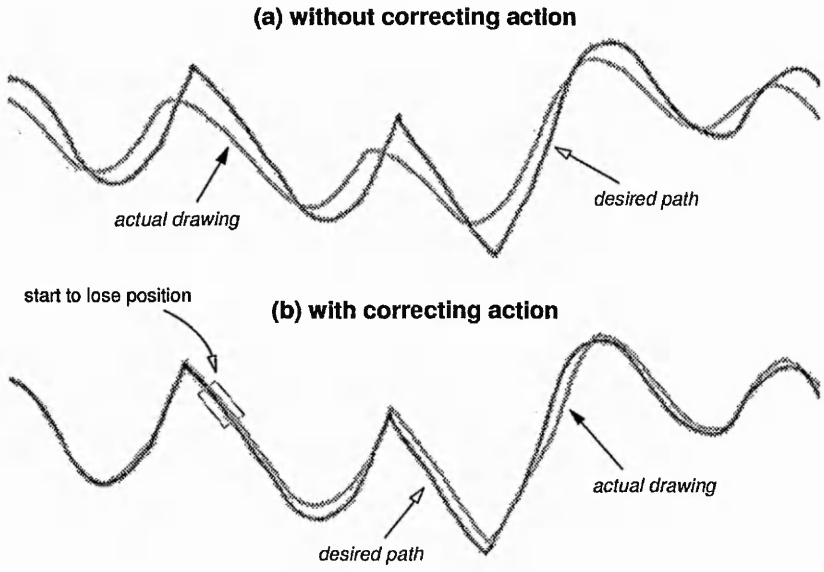


Figure 5-46: *Example C, using 3VMethod to correct the error, the SMP lost its position after a sharp angle of curve*

error still cannot be completely recovered. From observing a human operation controls the SMP, we found that the X axis of the machine has to be moved backward when a sharp angle curve is followed (marked by circles in Figure 5-45).

The researcher has learned two important factors from this investigation. First, the correction of the SMP error must be made in both X and Y coordinates, not only in one direction. Second, it is better that the correction made for each of the segments in the path is self-reliant. This will be a great help to the researcher for constructing the system - it is easier to correct only one error in a segment at a time rather than several errors from different segments together. A novel method has been designed based on this idea. Consequently an experimental system has been developed which yields excellent results. This is described in the next section.

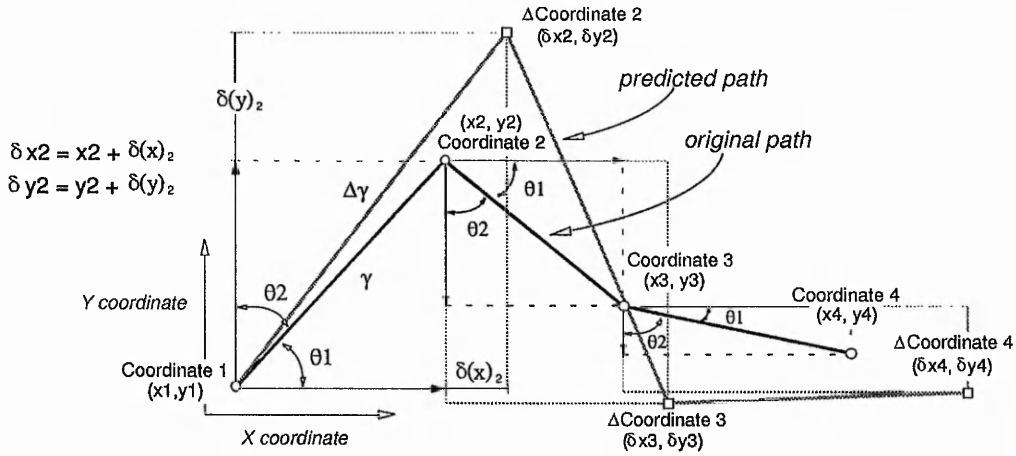


Figure 5-47: *Calculating a new corrected vector from* θ1 *and* θ2 *using the 2VMethod*

5.7.3.3 Two-vector Method

Similar to the 3VMethod, each time, this approach only manipulates a small portion of the desired path to predict the correction. Every *two* consecutive coordinates (a segment) are analysed over the entire path. As a straight line is connected from *Coordinate 1* to *Coordinate 2* (refer to Figure 5-47), the angles (θ1 and θ2) between the line (γ) and the X / Y coordinates are used to compute the possible correcting energies *(*δ(*x*) *and* δ(*y*)*)*. Figure 5-47 represents the calculation. The angles θ1 and θ2 are related to the each other - θ1 and θ2 are complementary. These two angles are passed to an A.I. engine which is designed by using inexact algorithms to determine the correcting energies (δ(x) and δ(y)). The prediction of the new estimated coordinate *(ΔCoordinate2(*δ*x2,* δ*y2))* is calculated by Equation 5-15. Equation 5-16 describes the procedure of computing the pattern of the predicted path.

$$\delta x(i) = x(i) + \delta(x)_i, \quad \delta y(i) = y(i) + \delta(y)_i \tag{5-15}$$

$$\begin{aligned} Predicted\,Path &= Coordinate(1) + \sum_{i=2}^{n} \Delta Coordinate(i) \\ &= \{x1, y1\} + \sum_{i=2}^{n} \{\delta x(i), \delta y(i)\} \end{aligned} \tag{5-16}$$

where i is the i^{th} segment of the path and n is the number of coordinates in the path.

Three different techniques based on the inexact algorithms, such as fuzzy logic, neural fuzzy theory, and neural networks have been applied to determine the correcting energies. In the following sections, the author presents a novel learning approach - the *Piecewise Error Compensation Algorithm (PEC Algorithm)* developed for solving the SMP following problems.

5.7.3.3.1 Fuzzy inference process

As the desired path is scanned by the Pre-PVS, the path is vectorised into 226 points. Two A.I. engines (one-input / one-output fuzzy inference system) are employed to predict the correcting energies which are divided into two parts: 1) *pattern of correction*, and 2) *amplitude of correction*. Both input and output fuzzy membership functions are divided into six overlapped fuzzy sets (NL, NM, NS, PS, PM, and PL). Symmetric triangular input function and symmetric trapezoid output function are applied to construct the fuzzy membership functions. The inference rules are listed in Table 5-5, where P is Positive, N is Negative, L is Large, M is Middle, and S is Small.

This system is mainly divided into two functional blocks. First, the *fuzzy engine one* reads the path-following-error (difference between paths) from the Post-PVS and decides a possible *amplitude of correction*. Then a new scanned path from the Pre-PVS is fed into the

IF	Angle is	NL	THEN correction is	NL
IF	Angle is	NM	THEN correction is	NM
IF	Angle is	NS	THEN correction is	NS
IF	Angle is	PS	THEN correction is	PS
IF	Angle is	PM	THEN correction is	PM
IF	Angle is	PL	THEN correction is	PL

Table 5-5: *Fuzzy rule base*

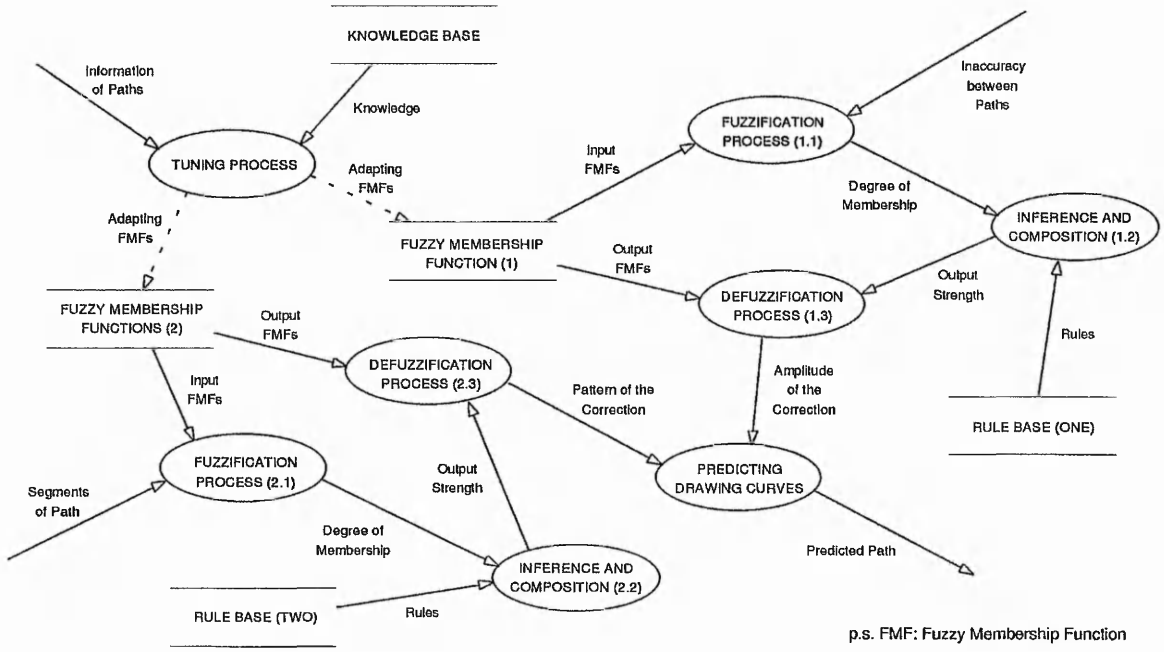


Figure 5-48: *Use of fuzzy engines to predict a compensated (predicted) path*

fuzzy engine two to calculate the *pattern (shape) of correction* (Figure 5-48). Equation 5-17 describes the process of combining these two data sets to produce the segments of the predicted path.

$$Predicted\ Segment(i) = Pattern_{segment(i)} \cdot Amplitude_{path} \qquad (5\text{-}17)$$

Figure 5-49 shows the initial setting of the system input and output membership functions. By applying this setting into the fuzzy engine, the system can produce a set of linear outputs. In order to adjust the fuzzy response to qualify the requirements of the system, a *tuning process*, which is illustrated in Figure 5-48, is devised to tune the fuzzy output membership function. As a result of various experiments, it was noticed that by adjusting a small part of the data points in the fuzzy output membership function (symmetric trapezoid), we can tune the response of the system into a desired pattern.

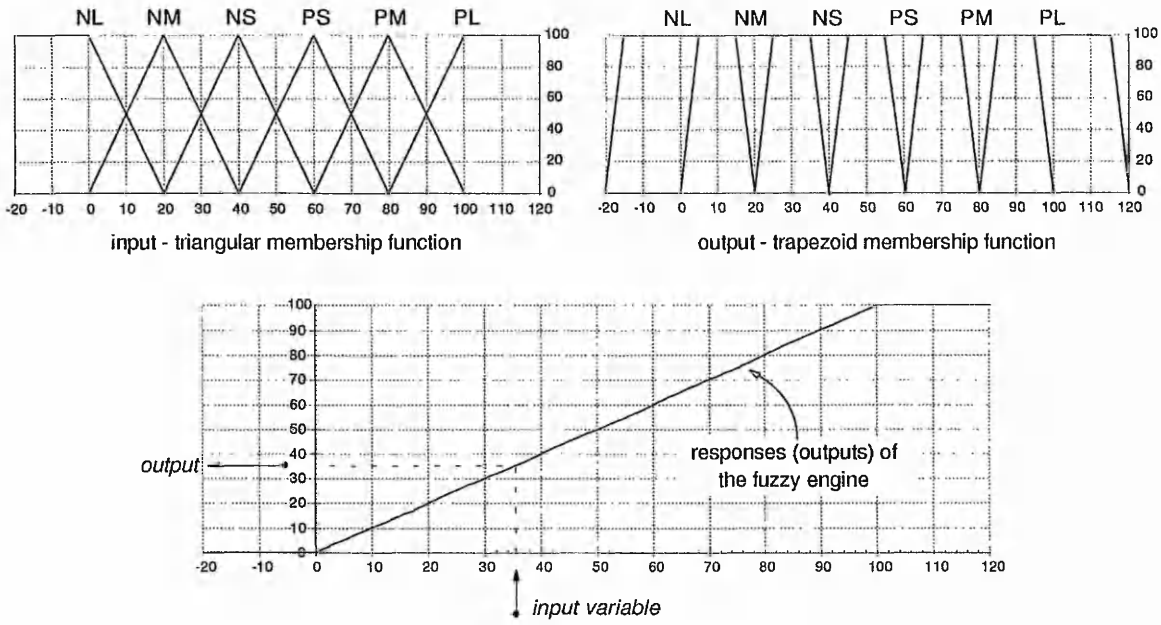


Figure 5-49: *System input / output membership functions and the fuzzy output pattern*

•The tuning process

As depicted in Figure 5-49, the membership functions have to be transformed into a set of data (input-configuration) for implementing the fuzzy engine. Figure 5-50 shows the configuration file of the output membership function. Since the symmetric trapezoid output membership functions are utilised to construct the engine, four data points are required to present a fuzzy variable. In order to simplify this procedure, only two data points S1 and S4

output membership function configuration file

fuzzy set	S1	S2	S3	S4
NL	-20	-15	15	20
NM	0	5	35	40
NS	20	25	55	60
PS	40	45	75	80
PM	60	65	95	100
PL	80	85	115	120

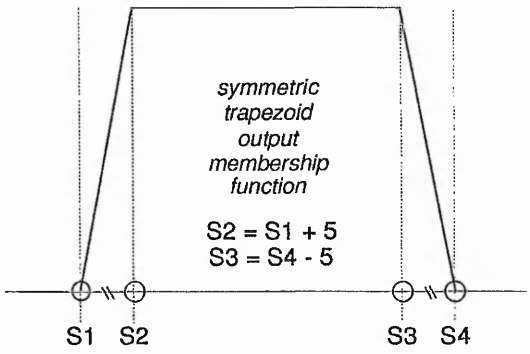


Figure 5-50: *Output membership function and its input-file*

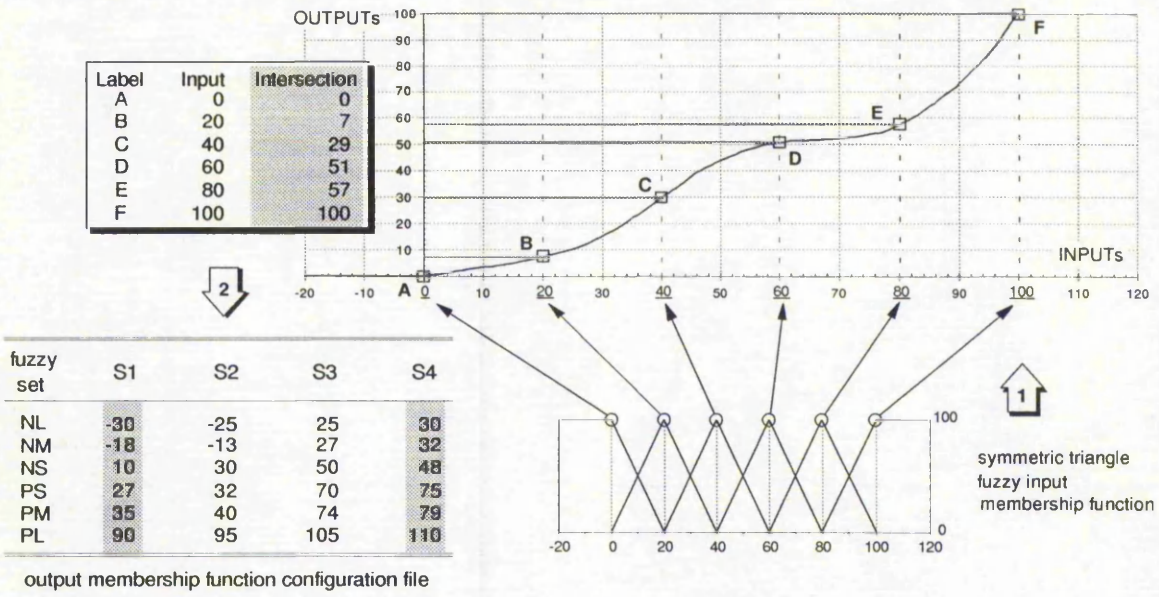


Figure 5-51: *Computing the fuzzy output membership function*

are used and S2 and S3 are calculated as follows: S2 = S1 + 5 and S3 = S4 -5 (Figure 5-50), where ± 5 are taken from the experiments.

Assuming that a fuzzy engine is applied to create a set of non-linear output data, as shown in Figure 5-51. Mapping the centre points of the fuzzy sets into the input membership function, six *intersection points* (A, B, C, D, E and F, labelled in Figure 5-51) can be determined. By manipulating these points and providing the outputs to Equation 5-17, an initial configuration of the output membership function (S1 and S4) can then be obtained.

$$(S1+S4)_{\text{fuzzy set}} \Big/ 2 = (Intersection\ Point)_{\text{fuzzy set}} \qquad (5\text{-}17)$$

The shape of the output pattern (curve) can be modified by selecting different values of S1 and S4. For instance, in Figure 5-51 at intersection point C, two sets of data: S1 = 5 & S4 = 53, and S1 = 10 & S4 = 48 are both satisfied in Equation 5-17 ((5 + 53) / 2 and (10 + 48) / 2 are both equal to 29). The shape of the output pattern between points B and D in Figure 5-51 is adapted by means of different values of S1 and S4 at point C. This process is heavily

dependent on designers' experiences. A significant amount of time is also needed to properly tune the membership functions before a solution is obtained. An example of configuration of the system output membership function which can be used to produce the non-linear fuzzy outputs is listed in Figure 5-51.

As stated previously, a corrected segment is equal to a predicted *pattern* of the segment times the predicted *amplitude* of the path. As the paper strip is passed to the *Post-PVS*, fuzzy engine one detects the path-following-errors and determines the *amplitude* of the correction. Also the *Pre-PVS* scans and analyses a new path on paper and provides the vectors to fuzzy engine two where the *pattern* of the correction is created. Combining the *pattern* and the *amplitude*, a predicted path can be constructed. This approach is divided into two functional blocks: *fuzzy pattern prediction* and *fuzzy amplitude prediction.*

a) Fuzzy pattern prediction

Using the technique stated in Figure 5-47, two system variable θ1 and θ2 can be

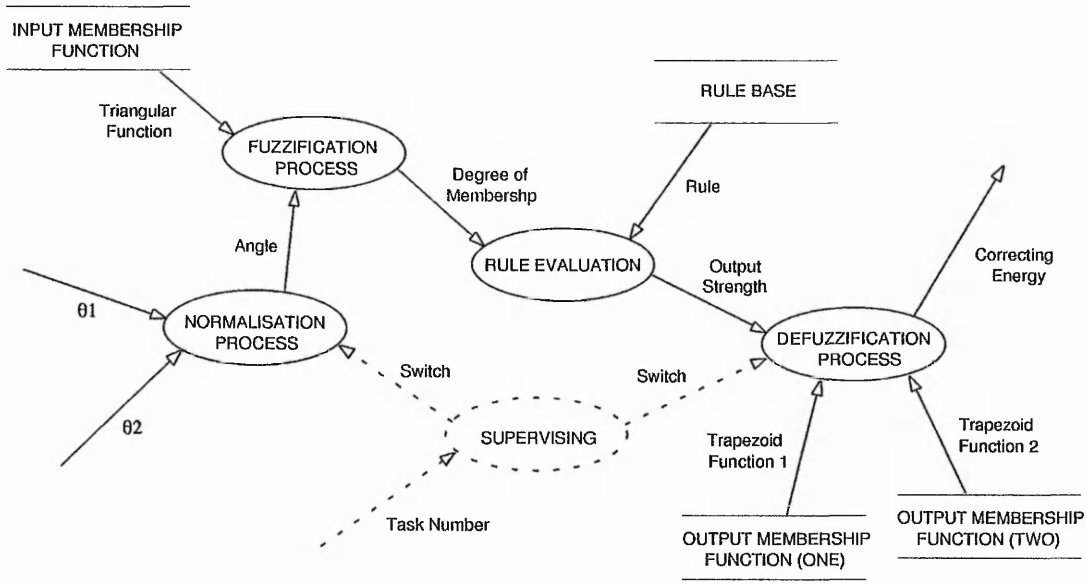


Figure 5-52: *Fuzzy engine two for determining the correcting energies*

obtained. These variables are provided to the fuzzy engine to determine the compensated pattern. The tuning process is engaged to configure the fuzzy system. The engine uses the same *input membership function* and the *rule base*. However, two different system *output membership functions* are provided to the system (Figure 5-52) where two different output patterns can be generated. The engine reads the first input variables (θ1), and *output membership function one* is used for to the defuzzification process. The correcting energy ($\delta(x)_i$) can be found. Applying the same approach to the second input variable (θ2) where the *output membership function two* is applied, the corresponding correcting energy ($\delta(y)_i$) can also be produced.

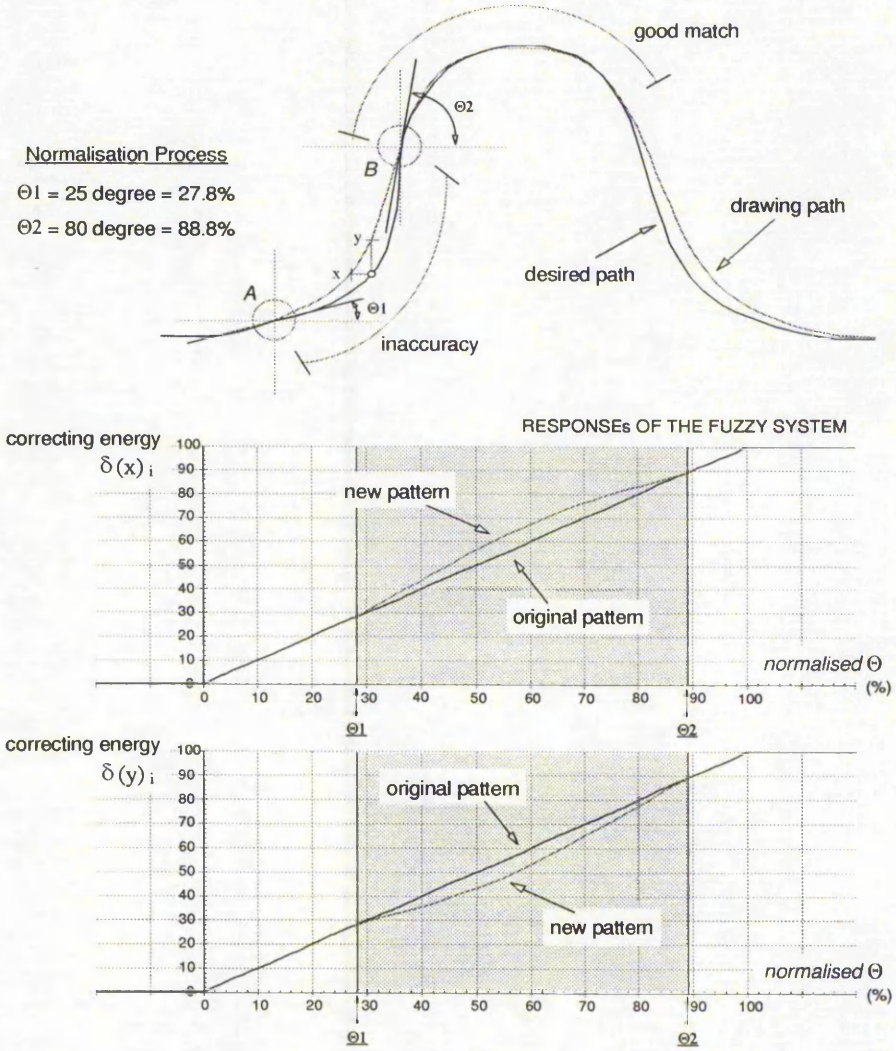
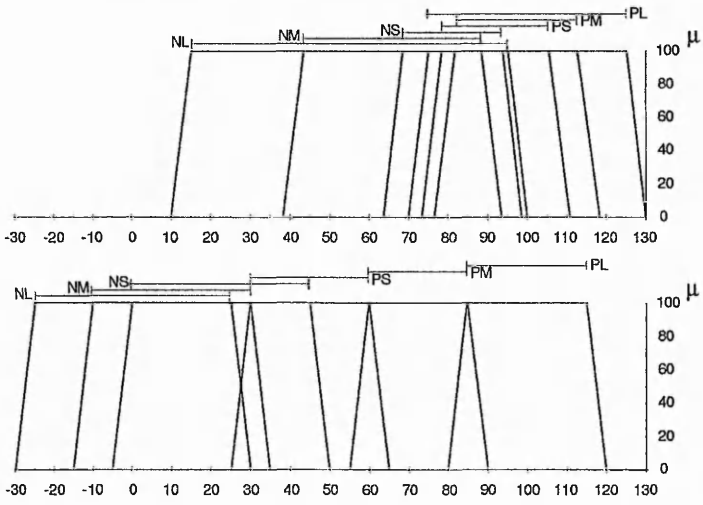


Figure 5-53: *Example of tuning the fuzzy system's response patterns*

output - trapezoid membership functions



membership function configuration files

for correcting energy $\delta(x)_i$

NL	10	15	95	100
NM	38	43	88	93
NS	63	68	93	98
PS	73	78	106	111
PM	77	82	113	118
PL	70	75	125	130

for correcting energy $\delta(y)_i$

NL	-30	-25	25	30
NM	-15	-10	30	35
NS	-5	0	45	50
PS	25	30	60	65
PM	55	60	85	90
PL	80	85	115	120

Figure 5-54: *The results of tuning the output membership functions*

Various experiments are undertaken to tune the output membership function into the desired pattern. As illustrated in Figure 5-53, a bell shaped path is used to test the response of the fuzzy system.

The system is initially configured by means of the rules mentioned in Table 5-5 and the membership functions in Figure 5-49. Carefully inspecting the paths in Figure 5-53, two sections of path-following-error are observed. The drawn path (using SMP) starts to lost its position at location A (marked by a circle), and re-matches with the desired path at location

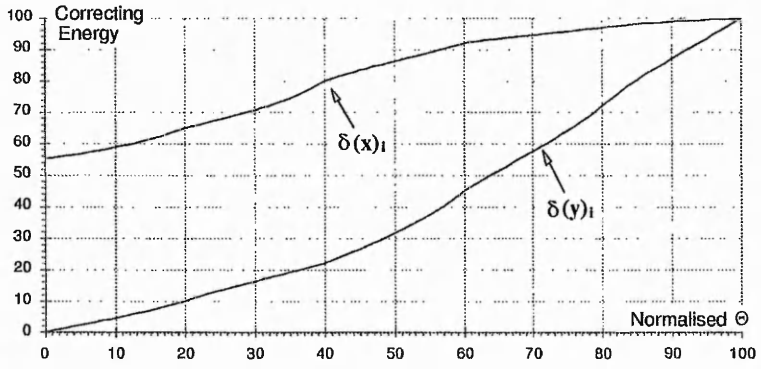


Figure 5-55: *Output patterns of the fuzzy system*

B. The system calculates the angles of the path at the position A and B. After normalising angles Θ1 and Θ2, the regions of the output patterns between these two data sets (the darker areas, as depicted in Figure 5-53) are modified.

The system has to increase the corrections in X direction and decrease in Y direction. The adapting procedures undertaken in the project are based on the trial and error method, and are highly dependent on engineers' experiences. Once the patterns of the system outputs are decided, the *tuning process* (described in the previous section) is applied to alter the *output membership functions*. Figure 5-54 and Figure 5-55 represent a sample of the tuned output membership functions and the fuzzy output patterns ($\delta(x)_i$ and $\delta(y)_i$). Two example paths using the fuzzy 2VMethod are shown in Figure 5-56.

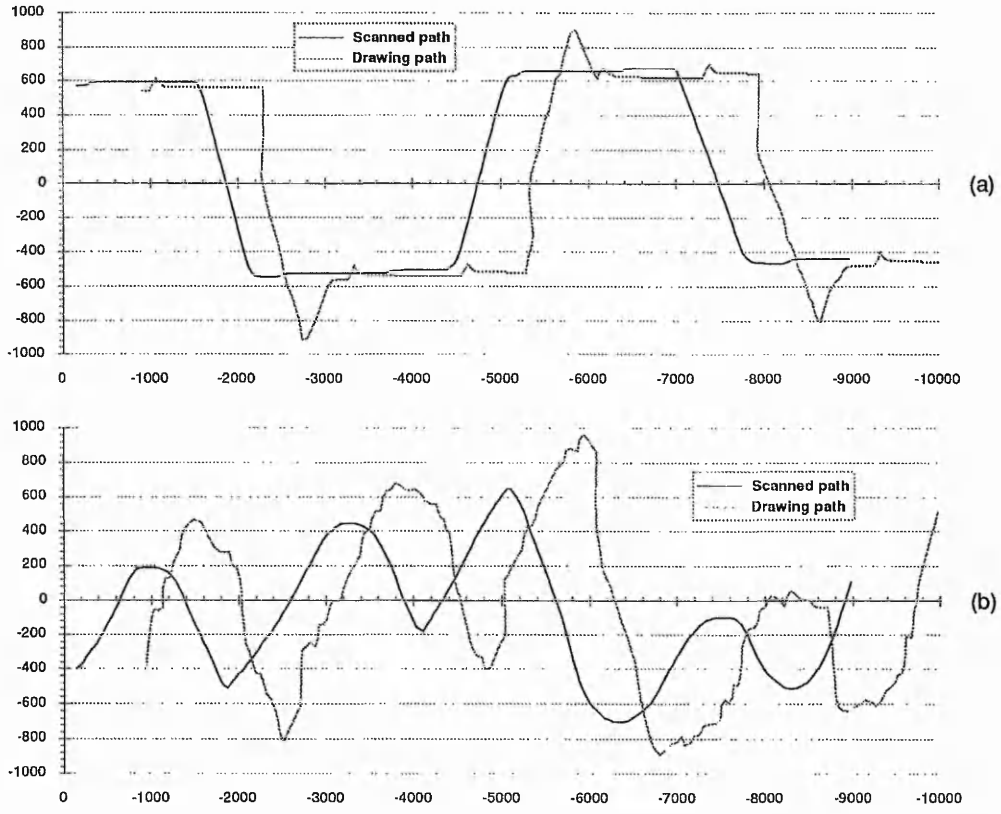


Figure 5-56: *Examples of finding the compensated path using the fuzzy 2VMethod*

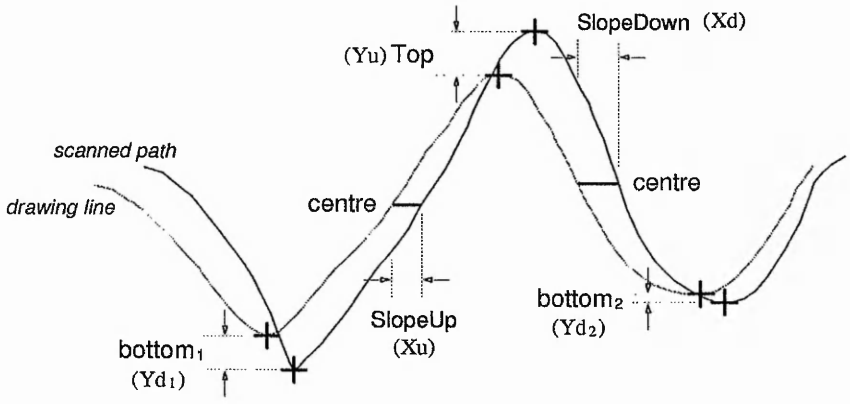


Figure 5-57: *Detecting the inaccuracy of path following*

b) Fuzzy amplitude prediction

Once the pattern of correction is determined, the next step is to decide the amount of the amplitude needed for the correction. As the first processed frame is passed under the Post-PVS, an image is taken and sent to the host system. The *colour curve differentiating technique* is employed to distinguish two different colour lines. The top, bottom and centre positions in both paths are taken to measure the inaccuracy of the SMP following. The distances between these points within the different paths (Figure 5-57) are calculated and passed to an A.I. Engine, such as a fuzzy, neural networks, or a neural fuzzy system. The engine takes the data and calculates the average errors for each of the parameters - *Top, Bottom, SlopeUp,* and *SlopeDown.* For instance, in Figure 5-57, *Bottom* = $(bottom_1 + bottom_2) \div 2$. Moreover, these parameters are used by the A.I Engine to determine a suitable amount of amplitude for the correction.

Figure 5-58 illustrates the fuzzy engine used to generate the updated amplitudes (*Yu, Yd, Xu,* and *Xd*) of the correction. The system reads four parameters - Top, Bottom, SlopeUp, and SlopeDown, respectively. Each of the parameters is passed to the fuzzy engine to decide a set of correcting amplitudes. Combining the new obtained amplitudes and the old data, the updated amplitudes can finally be attained. Using a configuration similar to *Fuzzy Engine*

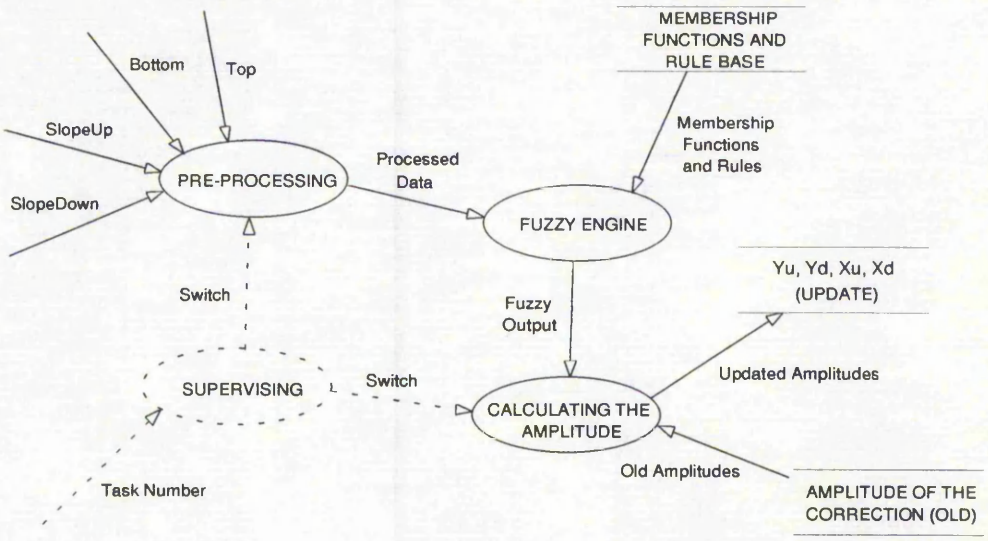
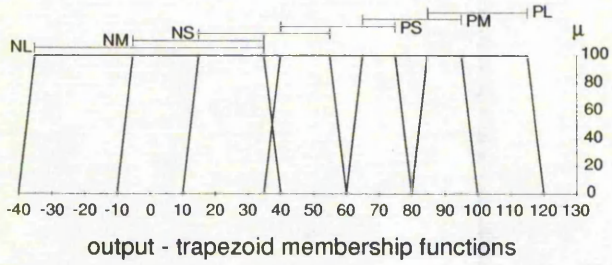


Figure 5-58: *Detecting the amplitudes of the correction using the PEC Algorithm*

Two described previously, the symmetric triangle input and trapezoid output membership functions are engaged to construct the *Fuzzy Engine One*. As indicated in Figure 5-59, a different system output membership function is employed to configure the engine. This inference system can produce a non-linear output pattern similar to the sigmoid transfer function, which is used to compute the correcting amplitudes. Figure 5-60 illustrates the drawing paths applied two predicted amplitudes of correction during the learning process.

5.7.3.3.2 Neural fuzzy inference process

In this approach, two *Neural Fuzzy Engines* have been assigned to replace the fuzzy



output membership function configuration file

fuzzy set	S1	S2	S3	S4
NL	-40	-35	35	40
NM	-10	-5	35	40
NS	10	15	55	60
PS	35	40	75	80
PM	60	65	95	100
PL	80	85	115	120

Figure 5-59: *System output membership function for Fuzzy Engine One*

	Amplitudes of Correction			
	Yu	Yd	Xu	Xd
Scanned Path	0	0	0	0
Drawing Path 1	-420	260	-100	-140
Drawing Path 2	-970	800	-450	-480

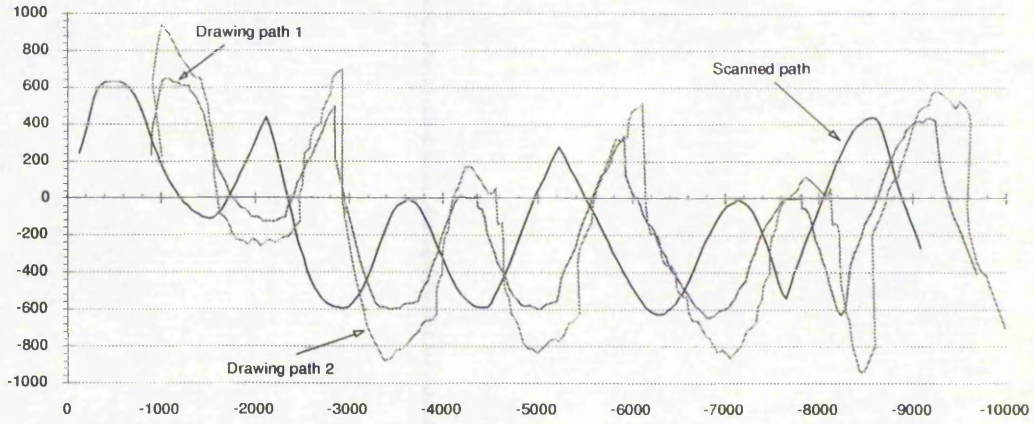


Figure 5-60: *Example of applying different amplitudes to generate the compensated path*

engines described in the last section. Instead of using the fuzzy membership functions and rules, the neural fuzzy system makes use of neural network for forming the required membership functions and the rule base. Using the same algorithms to correct the path-following-error, Neural Fuzzy Engine One determines the amounts of amplitudes for the correction. Besides, the Neural Fuzzy Engine Two constructs the correction pattern. Combining these two data sets, the predicted drawing path can be formed.

Similar to the *tuning process*, the *weight bases* (weights_a and weights_b labelled in Figure 5-61) of the neural fuzzy system are adjusted and tested by a *neural fuzzy training subsystem*. The system is divided into three functional blocks: obtaining the training data, training the network and testing the trained network. As depicted in Figure 5-61, the system updates (tunes) the *weight bases* to form the required system membership functions. After certain iterations of training, the system tests the outputs of the engine. The weights are continuously updated until the outputs of the system match the desired patterns. As the neural fuzzy system is learned successfully, it is employed to form the compensated drawing path. Figure 5-62 illustrates the data flow diagram for the system overview.

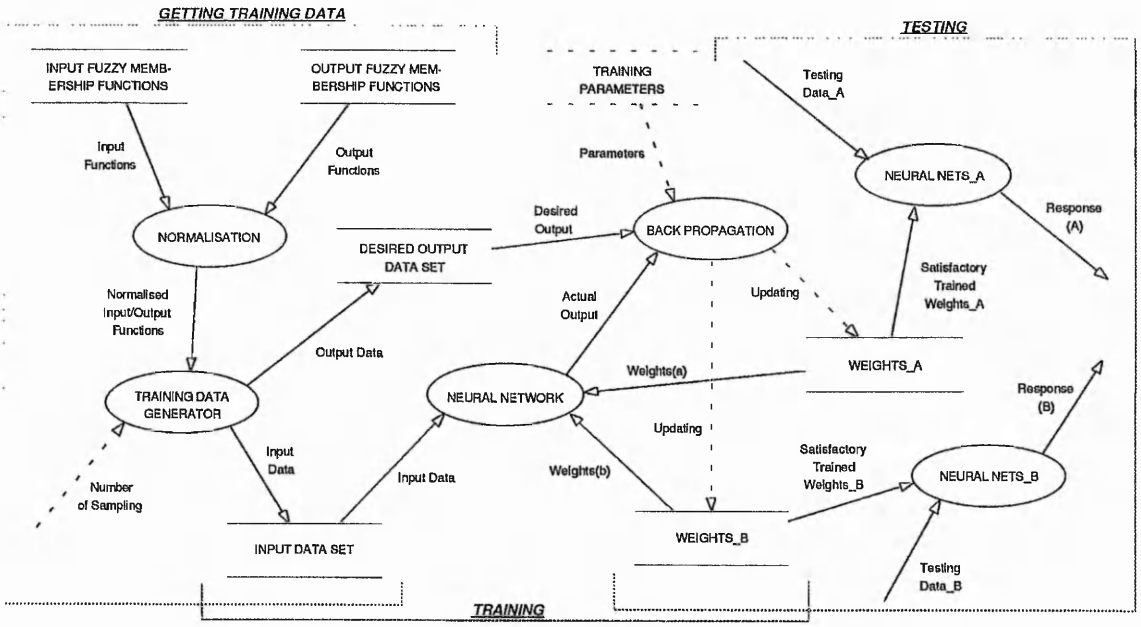


Figure 5-61: *Neural fuzzy training subsystem*

Two types of network architecture of the neural fuzzy system based on the techniques described in Section 5.3.4 are constructed. Layer 1 and layer 2 of the networks act as a fuzzification process, where Type A architecture (Figure 5-63) is fully connected structure

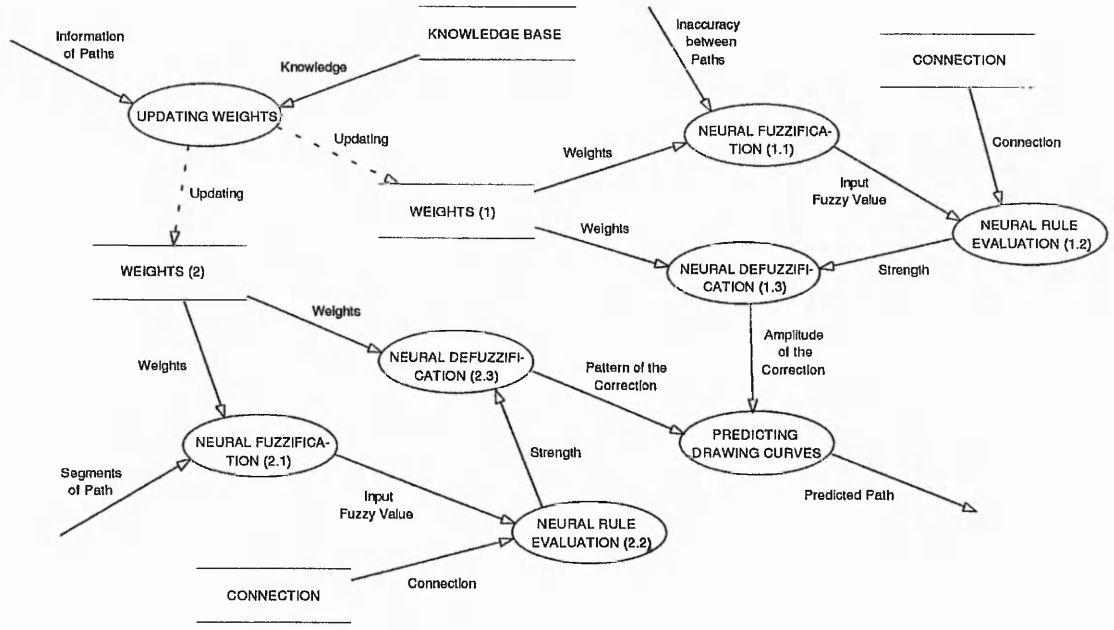


Figure 5-62: *Use of neural fuzzy engines to generate a compensated path*

and the Type B network (Figure 5-64) is partially connected. Both of the networks can successfully produce the desired outputs. According to the experimental results Type B architecture needs less converging time to train the network than the Type A nets. Nevertheless, since Type B structure uses more neurodes, it takes a little bit longer time to recall (use) the trained network.

An example of the system input and output membership functions generated by this novel neural fuzzy architecture is represented in Figure 5-65. It can be seen that the symmetric triangle and trapezoid membership functions can be successfully created by the network. The combination of these two types of membership functions can enforce the accuracy of the neural fuzzy system responses correctly.

5.7.3.3.3 Neural network inference process

The objective of this approach is to use a fuzzy system to supply the training data for a

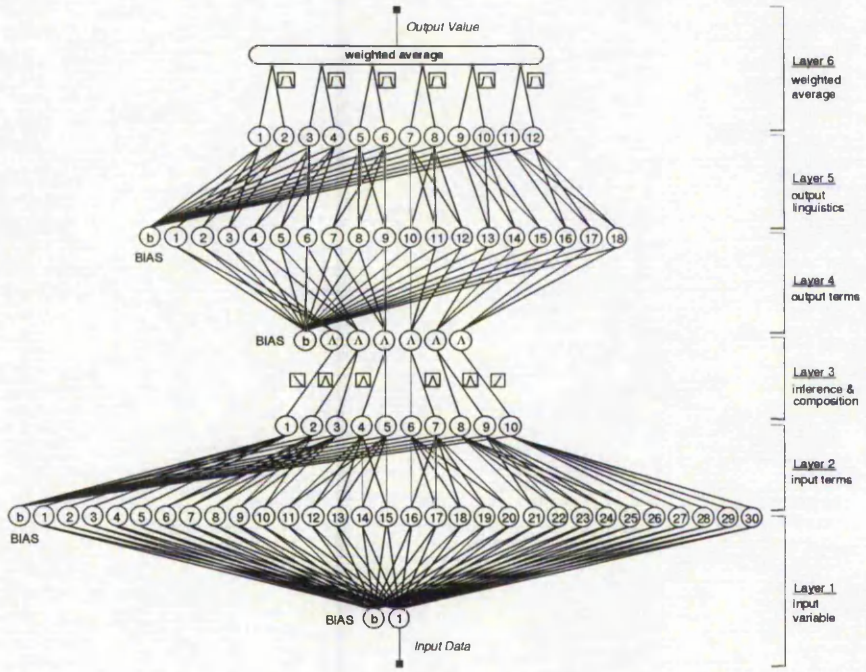


Figure 5-63: *Type A neural fuzzy architecture*

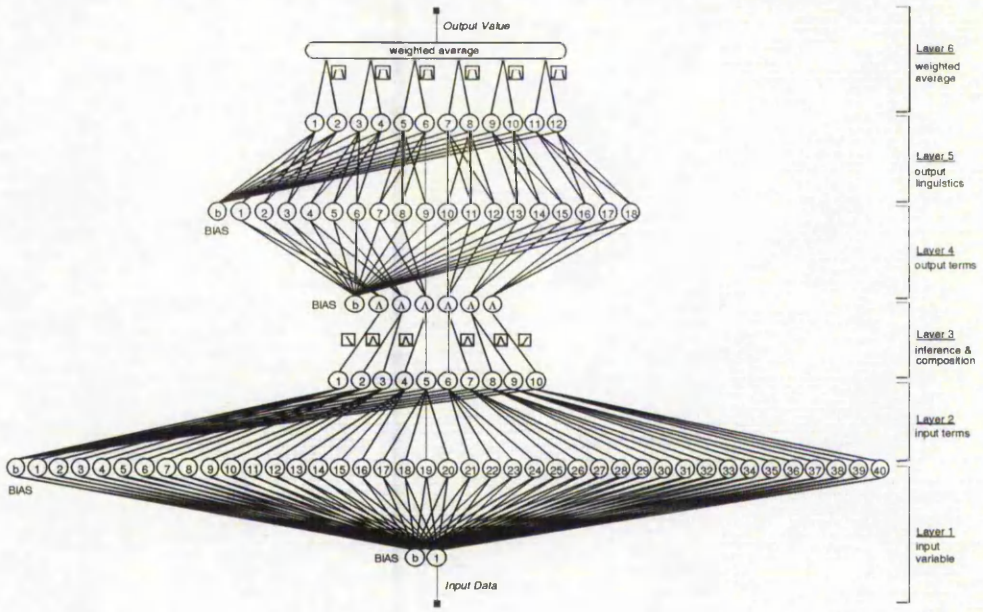


Figure 5-64: *Type B neural fuzzy architecture*

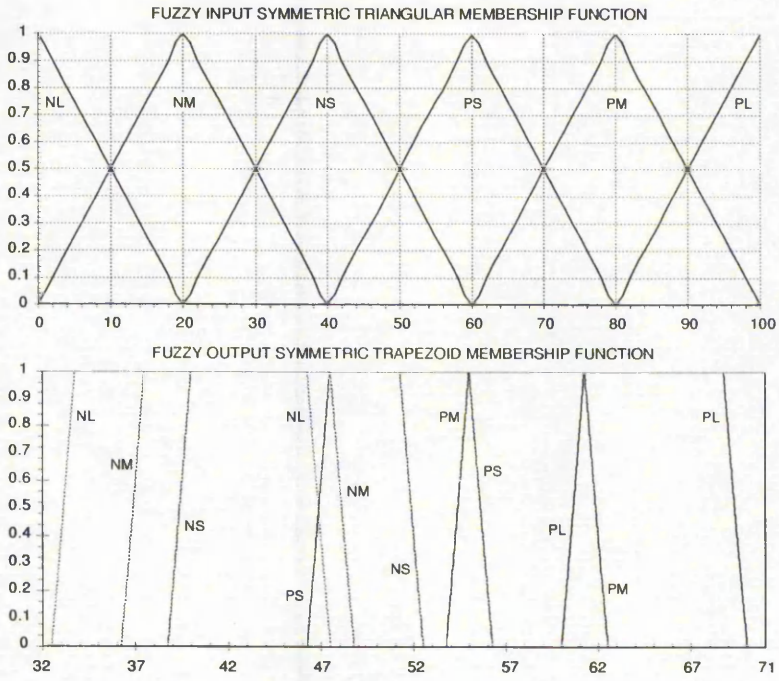


Figure 5-65: *Membership functions generated by the neural fuzzy engine*

multi-layered neural network and use the trained network to generate the correcting information to eliminate the SMP following error.

The network is trained with the supervised back-propagation algorithm on the input / output data pairs created by a fuzzy engine until it has learned the output actions determined by the fuzzy system, e.g. it basically becomes a "clone" of the fuzzy controller in the sense that its output behaviour imitates that of the fuzzy engine. The trained neural network is applied to guide the end effector to follow an irregular path.

The network constructed in the project is a fully connected three-layer back-propagation model comprising one input neurode, five hidden neurodes and one output neurode. The general architecture of the proposed system is shown in Figure 5-66. The flow of the process is in three steps. First, obtain a set of training data from the fuzzy engine. As the fuzzy system (described in Section 5.7.3.3.1) has been successfully designed (tuned), it is used to produce the training set for teaching the neural engine. Second, train the neural network. And, finally, test and recall the trained neural engine.

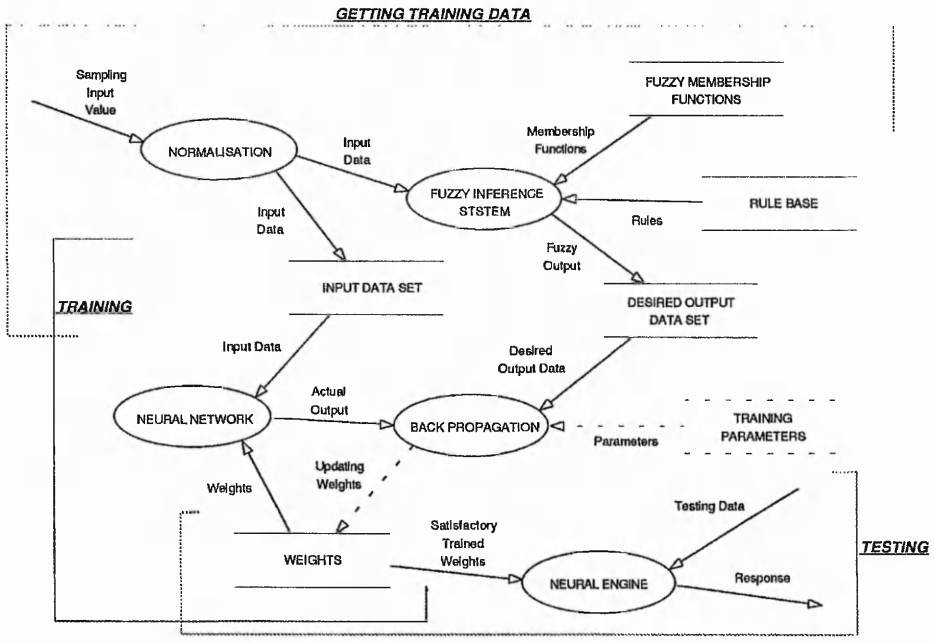


Figure 5-66: *General architecture of constructing the proposed neural network inference system*

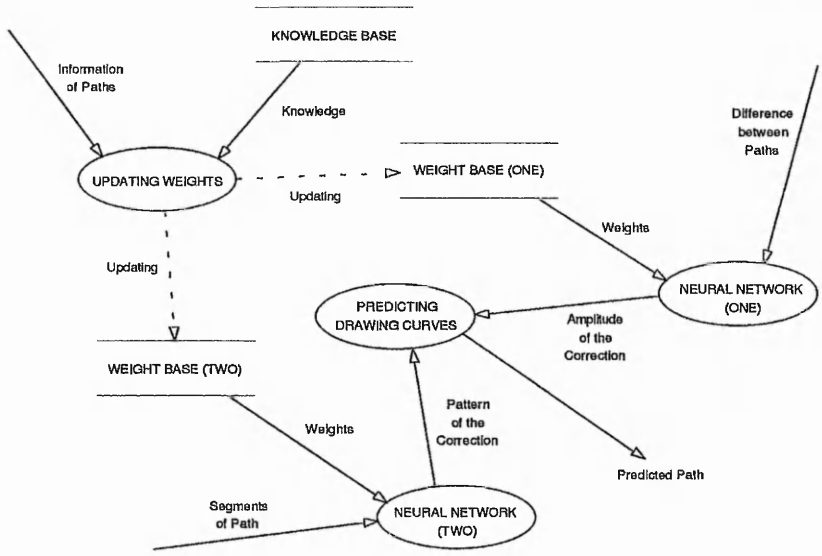


Figure 5-67: *Applying two neural engines to generate the compensated path*

As the network is successfully trained, the inference kernel reads the path-following-error and determines the amplitude of the correction. Besides, the desired path is fitted into the second neural engine which produces the pattern of the correction. Combining the outputs from these two engines, the compensated drawing path can be determined. Figure 5-67 illustrates the correlation among these tasks.

5.8 Experimental Results

Numerous experiments were carried out to evaluate the efficiency of this approach using the 2VMethod and the PEC Algorithm. Irregular shapes of curves are used in the testing. The Pre-PVS captured an image of the desired drawing path, the host system analyses the image and creates a compensated path for the frame. The Post-PVS grabs an image of the processed object. The path-following-errors between the *intended path* and the *resultant path* are used to decide a suitable amplitude for the correction.

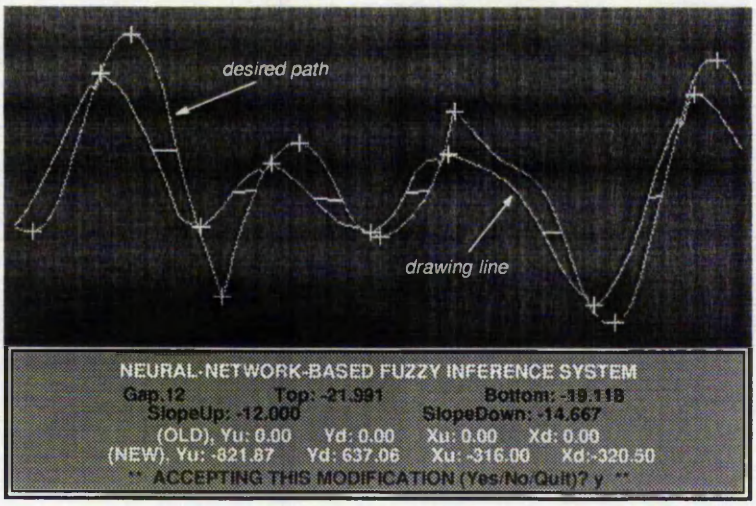


Figure 5-68: *Using the 2VMethod and the PEC Algorithm to determine the amplitude of the correction (Frame Zero)*

Figure 5-68 depicts the drawn line with no correction and its four predicted amplitudes (*New: Yu, Yd, Xu,* and *Xd*) for the correction. These four parameters are used to determine the compensated path for the subsequent frame. Figure 5-69 shows *Frame One* of the corrected path and its updated amplitudes. The similar process is continuously carried out until the two paths are matched together. Figure 5-70 and Figure 5-71 illustrate the Frame Two and Three of the correcting processes.

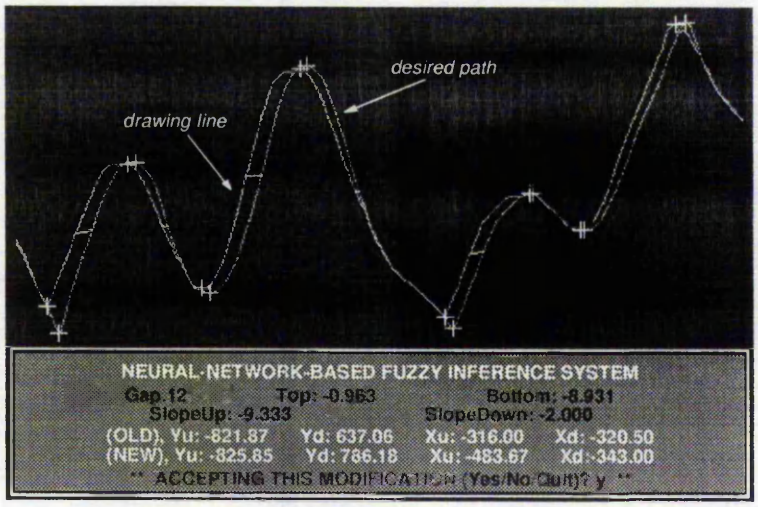


Figure 5-69: *Correcting process - Frame One*

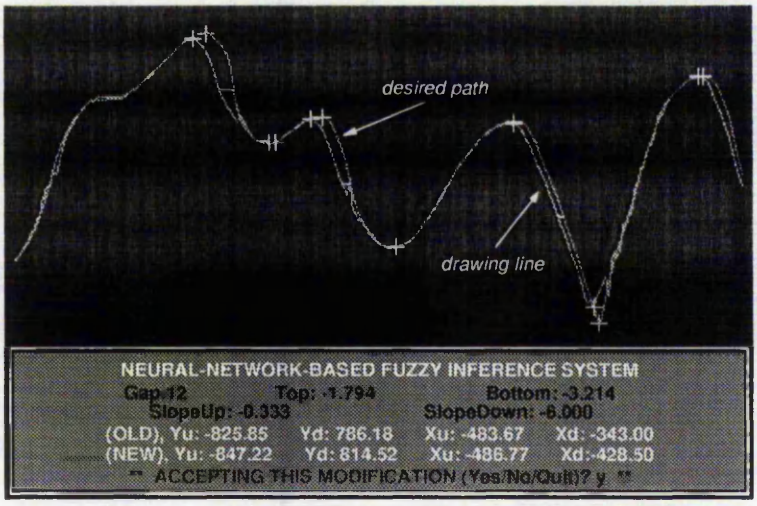


Figure 5-70: *Correcting process - Frame Two*

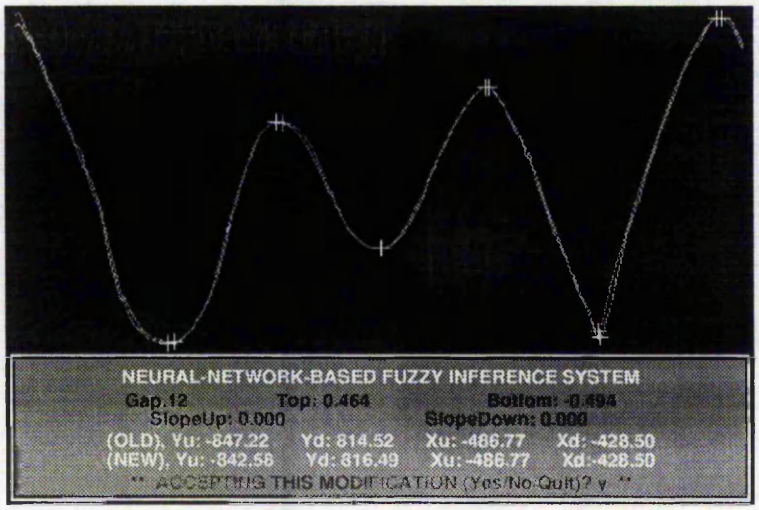


Figure 5-71: *Correcting process - Frame Three*

According to various experiments, after three frames of correcting processes almost all the path-following-error caused by the spring can be eliminated. In addition, the obtained (learned) control parameters can be utilised to correct the SMP following errors in the subsequent frames as well as dealing with any irregular shapes of curves. The results indicate that the *2VMethod* applied the *Piecewise Error Compensation Algorithm* only needs a few frames of self-learning processes to precisely correct for the path-following-errors caused by the SMP. Table 5-6 represents some samples of the learned control parameters

	Amplitudes			
Sample No.	*Yu*	*Yd*	*Xu*	*Xd*
1	-989.595	831.023	-426.614	-424.500
2	-978.326	809.895	-403.630	-403.630
3	-994.032	800.000	-414.500	-423.826
4	-1139.712	654.622	-481.951	-496.701
5	-965.462	594.649	-463.252	-448.141
6	-802.240	479.759	-293.050	-302.190
7	-915.919	451.220	-293.103	-285.083
8	-849.369	508.109	-340.063	-339.047
9	-977.224	525.947	-346.914	-350.083
10	-860.326	492.808	-305.640	-305.000

Table 5-6: *Examples of the learned amplitudes*

acquired from various training processes. It can be seen that once the Z axis of the cutting mechanism is reset (this will change the pressure applied to the SMP), the obtained system parameters Yu, Yd, Xu, and Xd are entirely altered. Various sample patterns of the SMP path following process using the 2VMethod together with the PEC Algorithm are illustrated in Appendix C.

5.9 Summary

This chapter discusses the *inexact algorithms* employed in the tightly coupled vision and control system. The system is mainly divided into two sections: pre-processing vision system (Pre-PVS) and post-processing vision system (Post-PVS).

A *spring mounted pen* (SMP) is engaged to emulate the movement of the lace strip due to the cutting force caused by the tactile cutter and the mechanical feed misalignment. Due to the flexibility of the dynamic structures (such as lace patterns and the SMP) and complexity of the correcting process, the inference methods based on *fuzzy logic, neural fuzzy technique*, and *neural networks* are applied to overcome the problems. Moreover, it is

proposed to use the inexact algorithm as the basis for achieving optimal quality which satisfies visual demands rather than engineering precision.

A number of approaches have been described attempting to correct the path-following-error. The experimental results indicate the capability and effectiveness of the proposed algorithms. The development of the system is a novel approach to martial processing and has further applications where deformable materials and structures are processed.

6. GENERIC ERROR COMPENSATION ALGORITHM

Chapter 6

GENERIC ERROR COMPENSATION ALGORITHM

6.1 Introduction

A learning method, the *Piecewise Error Compensation Algorithm (PEC Algorithm)*, developed for minimising the errors caused by the Sprint Mounted Pen was described in the previous chapter. The PEC Algorithm can learn to correct the system's behaviour in terms of analysing the differences (path-following-errors) between the intended patterns and the actual patterns. Approximately three frames of correcting process are required before the machine reacts correctly in minimising the deviation.

In order to increase the speed and the accuracy of the correcting process, an improved learning algorithm is developed. This algorithm is based on utilising the 2VMethod together with a new scheme for detecting the correcting pattern as well as the amplitude. Similar to the PEC Algorithm, the inexact algorithms are employed to construct the A.I. kernel. Equations 6-1 and 6-2 describe the process of computing a compensated path.

$$Compensated\ Path = \sum_{i=1}^{n} \{Compensated\ segment[i] + Original\ segment[i]\} \qquad (6\text{-}1)$$

$$\delta(x,y)_i = CP_{segment[i]} \times CA_{path} \qquad (6\text{-}2)$$

where n is the number of coordinates in the path, CP denotes the *Correcting Pattern*, CA is the *Correcting Amplitude* and $\delta(x,y)_i$ is the compensated segments. In the following sections, this new type of learning algorithm is described, and is divided into two functional blocks: *detecting the correcting pattern* and *detection of the correcting amplitude*.

6.2 Detecting the Correcting Pattern

In order to detect (obtain) the deviation caused by the spring, the SMP is driven to follow a template (a square wave) on paper. The image of this square wave is captured by

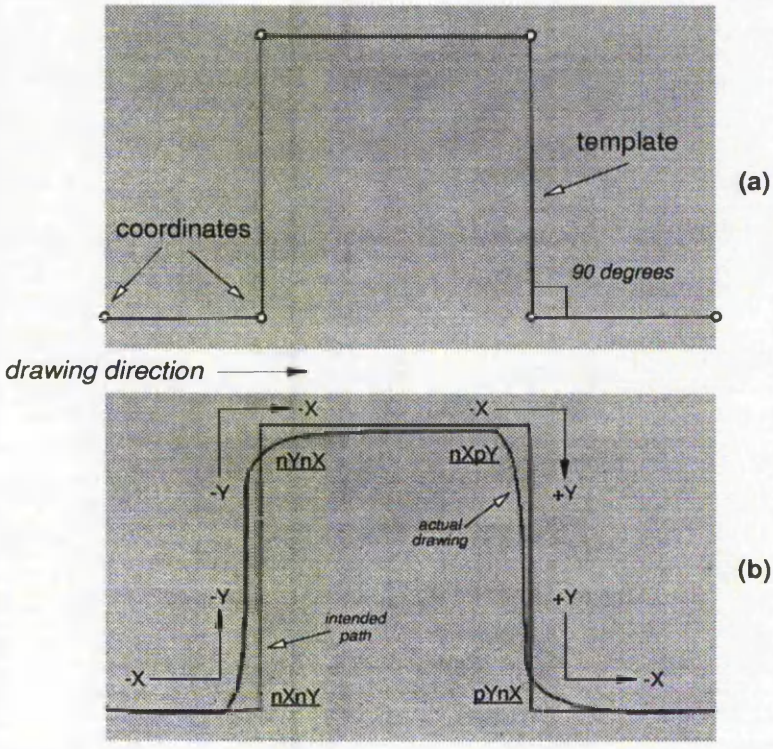


Figure 6-1: *(a) Vectorising the square wave; (b) Drawing a second path followed the template using the SMP*

the vision system. A software filter is developed to detect this captured image - six coordinates, such as shown in Figure 6-1 (a), can be obtained. This data is then transferred to the controller.

The SMP is driven to draw a second line on the paper. It is clear that the path-following-errors appear. The path-following-errors appear when the direction of the drawing changes, i.e. from direction -X to -Y, -Y to -X, -X to +Y and +Y to -X (see Figure 6-1 (b)). There are four different types of deviation patterns that can be detected. They are labelled as nXnY, nYnX, nXpY and pYnX where 'n' denotes negative and 'p' means positive.

The inexact algorithms are employed here for constructing the A.I. engines to produce the compensation patterns. The artificial neural network approach engaged to learn the correcting action from these deviation patterns is illustrated. Figure 6-2 depicts the use of the nXnY deviation pattern to produce the learning data set for training a neural engine (ANN Engine). Eleven data points (a, b, c, ..., k labelled in Figure 6-2) which are taken from experience are chosen in this instance.

Table 6-1 represents the training data pairs collected from the nXnY deviation

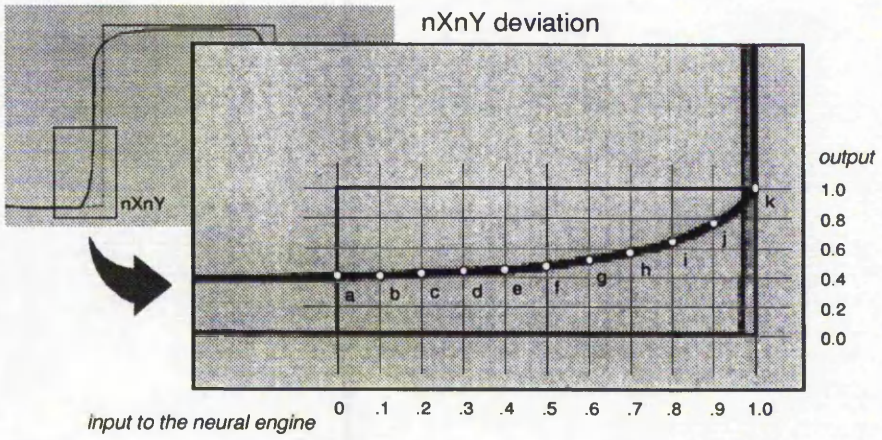


Figure 6-2: *Obtaining the training data set from a deviation pattern*

label	a	b	c	d	e	f
input	0.0	0.1	0.2	0.3	0.4	0.5
output	0.395	0.402	0.405	0.410	0.422	0.470
label	**g**	**h**	**i**	**j**	**k**	
input	0.6	0.7	0.8	0.9	1.0	
output	0.520	0.570	0.630	0.750	1.000	

Table 6-1: *The training data pairs collected from nXnY pattern*

pattern. As the training data is fed into the neural engine, which is a standard fully interconnected three-layer back-propagation network, after learning and updating procedure the trained neural engine can be used to generate a correcting pattern. Using the similar approach stated above, the learning data sets from nYnX, nXpY and pYnX deviation patterns can all be obtained and utilised for teaching the network. Figure 6-3 illustrates the processes and data sets used to produce the correcting patterns.

As mentioned previously, four different shapes of deviation patterns, i.e., nXnY, nYnX, nXpY and pYnX, can be detected from the result of following the template. Once

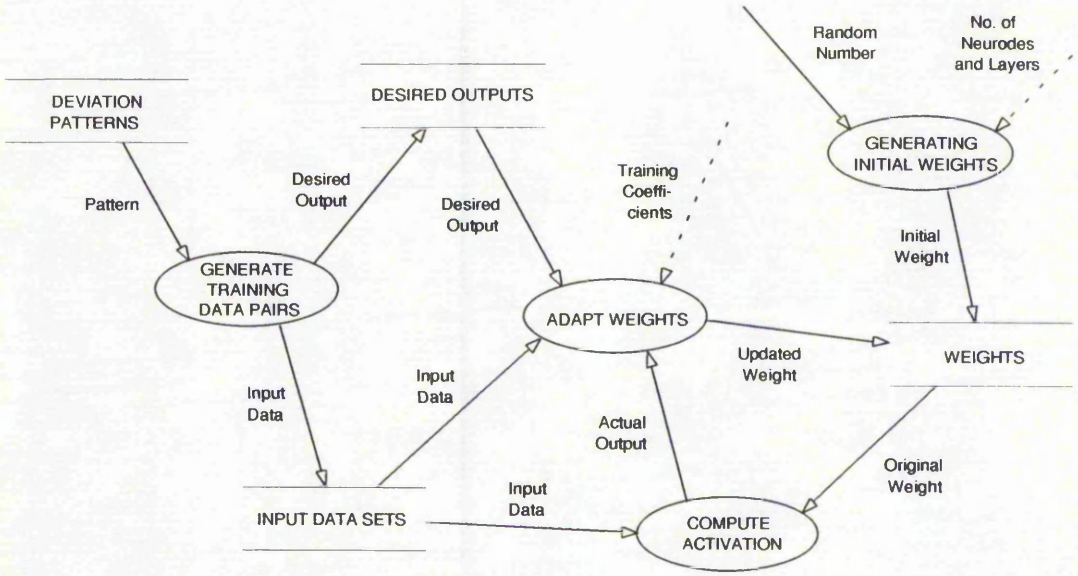


Figure 6-3: *The neural engine used to learn the correcting patterns*

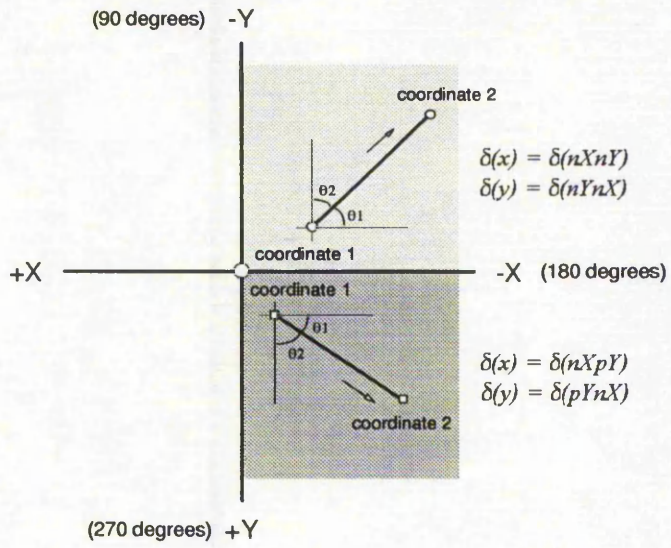


Figure 6-4: *Depending on the direction (angle) of the path, two sets of correcting patterns can be chosen to assign for the 2VMethod*

the neural engine has successfully learned from these samples to produce the correcting patterns, the 2VMethod is then, used to calculate the compensated segments. Dissimilar to the scheme mentioned in Section 5.7.3.3, instead of using only two correcting patterns (refer to Section 5.7.3.1: the tuning process, Figure 5-53), two sets of correcting patterns are used, i.e. $\{\delta(nXnY), \delta(nYnX)\}$ and $\{\delta(nXpY), \delta(pYnX)\}$. Depending on the direction (angle) of the line between two detected coordinates, one of the correcting pattern pair is assigned to the correcting energies $\delta(x)$ and $\delta(y)$. For example, if the angle of the line between two coordinates is less then 180 degrees (Figure 6-4), then Equation 6-3 is engaged by the 2VMethod; or if the angle of the line is larger than 180 degrees then Equation 6-4 is active.

$$\delta(x) = \delta(nXnY),\ \delta(y) = \delta(nYnX) \tag{6-3}$$

$$\delta(x) = \delta(nXpY),\ \delta(y) = \delta(pYnX) \tag{6-4}$$

where $\delta(nXnY)$ is the correcting pattern generated by the neural engine which uses $nXnY$ deviation pattern as the learning data. Since the cutting mechanism employed in the project

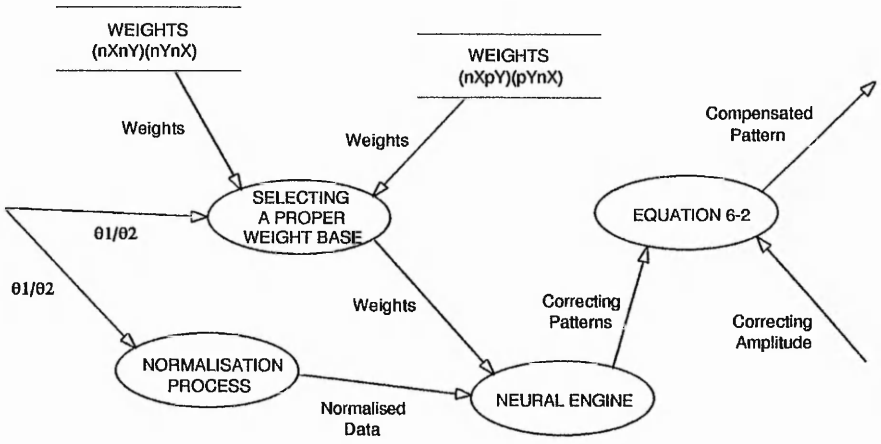


Figure 6-5: *Use of a neural engine to compute the compensated pattern*

is controlled to move from +X to −X direction, that is why we only need to consider the angles of the coordinates which are larger than 90 degrees and less than 270 degrees.

As the angles θ1 and θ2 (labelled in Figure 6-4) are detected, this information is normalised into the range of [0, 1]. Additionally, this normalised data is passed into the trained neural engine in order to produce the correcting energies $\delta(x)$ and $\delta(y)$ which can be used to create the correcting pattern by utilising the 2VMethod. Figure 6-5 represents this procedure.

6.3 Detection of Correcting Amplitude

Once the correcting patterns are obtained, the next step is to determine the amount of correcting amplitude required. As already mentioned while the Z axis of the testing rig is reset, three system parameters of the SMP are altered (refer to Section-6-2). This results in changing the magnitude of the path-following-errors (Figure 6-6). In order to measure the maximum amount of path-following-error that can be produced by the SMP, the actual length of the deviation is calculated.

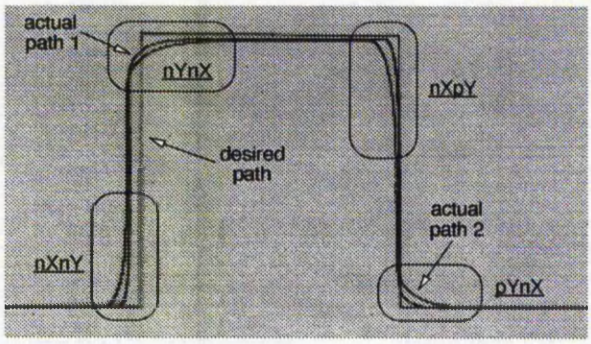


Figure 6-6: *Sample of template following using different system setting*

Figure 6-7 depicts an example of calculating two maximum deviations caused by the SMP. The vision system is used to detect the length of L1 (or L2). The actual length of L1 is then transformed into the machine control unit which is 40 steps / mm. As an illustration, if L1 is measured as 8.9 mm, the maximum machine control units can be added in the original path in $\delta(nXpY)$ side is 356 steps (8.9 mm × 40 steps / mm).

Since a low resolution vision station is engaged in the project, it is non-trivial to set up the system to detect and calculate the precise length of the deviation (L1 / L2 indicated in Figure 6-7). In order to improve this process, an alternative approach is developed. The

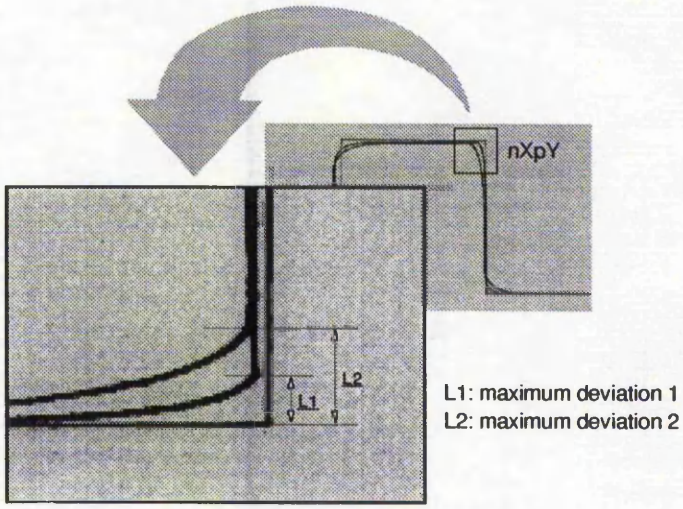


Figure 6-7: *Example of calculating maximum deviations*

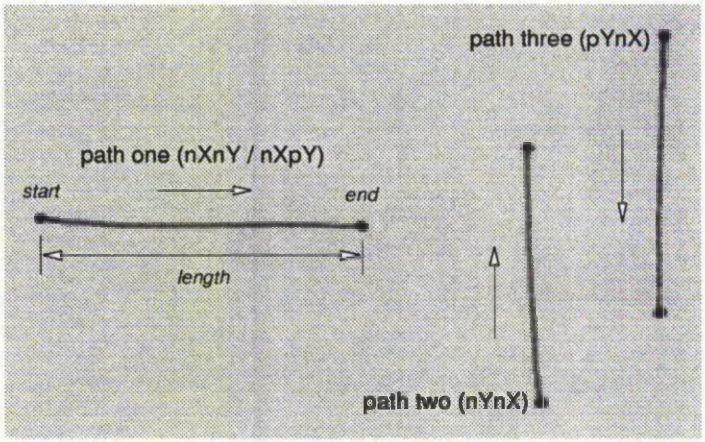


Figure 6-8: *Samples of calculating the lengths of the drawing lines*

SMP is driven to draw three short straight lines (six centimeter was used in the experiments) on paper, as shown in Figure 6-8.

The three lines were drawn on the white paper and captured by the camera. After correcting the optical distortion, the actual lengths of these lines are calculated. For example, *path one* in Figure 6-8 is calculated as 51.5 mm. And the cutting mechanism is actually moved 60 mm. Therefore we can calculate the deviation is 8.5 mm, and the maximum machine control units can be added in the original path in $\delta(nXnY)$ side is 340 steps (8.5 mm × 40 steps / mm). Noted that based on to various experimental results the researcher assigns that the maximum deviations in nXnY and nXpY sides are identical. This is why only three lines need to be drawn rather than four lines.

6.4 The Correction Process

As the testing rig is set up, the SMP is driven to draw a square wave on paper. Four deviation patterns are taken to create the learning data sets for training the neural network kernel. In addition, the testing rig is driven to draw three straight lines on white paper. The

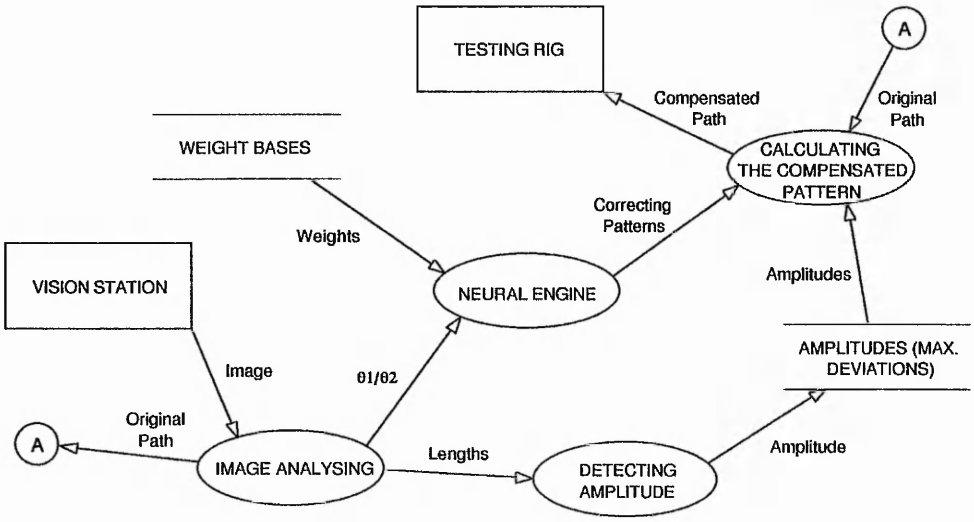


Figure 6-9: *DFD for the overview of the correcting process*

vision system then captures an image of the drawing. Three different maximum deviations (nXnY/nXpY, nYnX and pYnX) are detected and stored in a configuration file.

A new frame of the desired drawing path is grabbed by the camera and analysed. The detected pattern (original path) is then vectorised and fed into the trained neural engine (Figure 6-9). By using Equations 6-1 and 6-2, the correcting patterns and amplitudes as well as the original detected path are used by the 2VMethod to create the compensated path. This vectorised data is, finally, transferred to the controller. Figure 6-10 illustrates the detected correcting amplitudes and the compensated path generated by the neural engine.

6.5 Experimental Results

Various experiments were carried out to evaluate the effectiveness of this approach. Regular and irregular shapes of paths are used in the testing. The modified version of the 2VMethod and Equation 6-1 / 6-2 are used to create the compensated pattern. Figure 6-11 depicts the processes of correcting the path-following-errors using the *Generic Error Compensation Algorithm (GEC Algorithm).* Frame (a) in Figure 6-11 shows the amount of

	Amplitudes of Correction			
	nYnX	pYnX	nXnY	nXpY
Intended Path	0	0	0	0
Compensated Path	-754	623	-324.5	-324.5

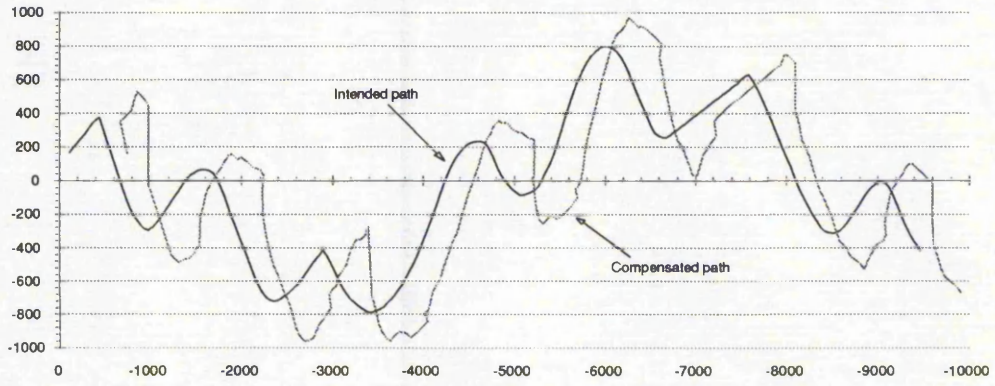


Figure 6-10: *Example of applying the neural kernel to create the compensated path*

deviation generated. After the learning process (Frame (b)) the maximum deviations are detected and stored in a configuration file. The intelligent machine console, then, takes the compensated patterns from the trained neural engine together with the maximum deviations to create the compensated path. Lastly, the cutting mechanism controls the SMP to draw a second line overlapping with the desired pattern (such as shown in Frame (c), Figure 6-11).

As mentioned previously, a low resolution vision station is engaged. While the system calculates the maximum deviations during the learning process, a certain amount of error is introduced due to camera distortions. Besides, the SMP is liable to vibrate while the testing rig retreats the pen to move to a new position - this will cause random positional errors. In other words, both situations stated above will strongly affect the precise detection of the maximum deviations. Consequently, when the obtained information is provided to the controller, the results of the SMP following process may be inaccurate.

Figure 6-11 (c) depicts the result of SMP following process using the neural network approach. As we can see, almost all the errors are successfully removed - the intended path

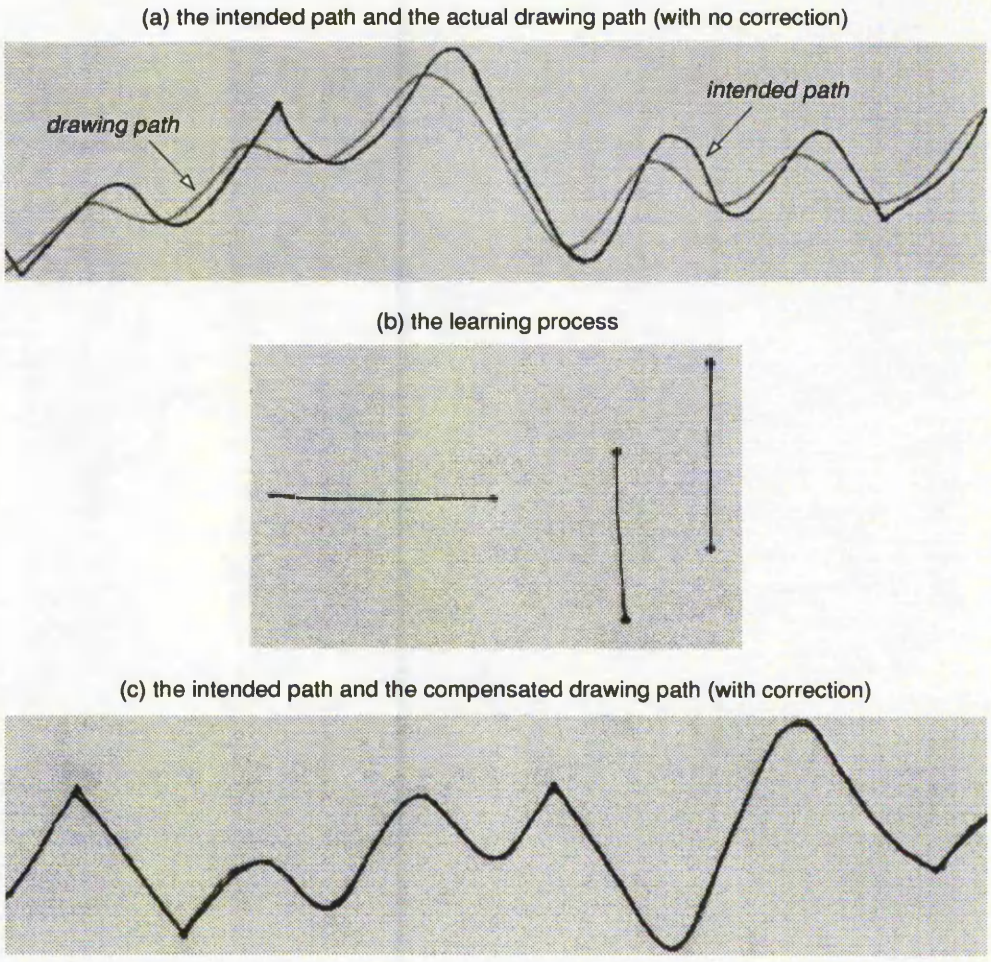


Figure 6-11: *The processes of correcting the path-following-errors using the Generic Error Compensation Algorithm (GEC Algorithm)*

and the actual path are matched together. A small amount of inaccuracy left between the paths is caused by the miscalculation of the maximum deviations. This can be easily worked out by means of a higher accuracy / resolution vision system and experimental set up.

In fact, if both *deviation patterns* and *the correcting amplitudes (maximum deviations)* can be precisely analysed and recorded by a high accurate sensing station, the entire path-following-errors caused by the spring can potentially be removed by applying this GEC Algorithm with the 2VMethod. More samples of SMP following process and their detected compensated patterns can be found in Appendix D.

6.6 Summary

This chapter describes a *Generic Error Compensation Algorithm* developed for correcting the path-following-errors generated by the SMP. In contrast to the previous chapter (the PEC Algorithm), only one frame of learning process is required before the system can correctly minimise the errors. The system is developed based on applying the inexact algorithm. The neural network approach was used to implement this algorithm.

A number of experiments were carried out to estimate the capability of this method. Any regular and irregular shapes of paths were used for testing the SMP following. According to the experimental results, using the algorithm developed the intelligent machine console can potentially remove all the errors due to flexibility of dynamic structures.

7. DISCUSSION

Chapter 7
DISCUSSION

7.1 Discussion

The previous chapters have described the various algorithms applied to develop a vision based intelligent control system to deal with problems in handling flexibility of dynamic structures, such as lace patterns and the spring mounted pen.

In order to compensate the errors due to these deformable structures, the inexact algorithms, namely fuzzy logic, neural networks, and neural fuzzy technique, have been employed in the system to construct an intelligent host controller. Using the pre- and post-processing vision stations with the intelligent controller, the system can automatically compensate for flexibility. The machine vision system developed consists of the following main components.

§ Lace Pattern Detection

- pixel intensity directed feature extraction (PIDFE);
- fuzzy pattern recognition technique (FPR);
- line mapping method (LMM);
- supervision of the system.

§ Correction of Cutting Operation

- colour curves differentiation;
- three-vector method (3VMethod);
- two-vector method (2VMethod);
- piecewise error compensation algorithm (PEC Algorithm);
- generic error compensation algorithm (GEC Algorithm).

In order to allow investigation of the algorithms, the software has been written by the author to measure the capabilities and effectiveness of the approaches. The development of the software has been carried out using an IBM compatible personal computer and the 'C' programming language (Borland C/C++ compiler, Version 3.1). The programes have been developed with a high degree of modularity. This makes it possible to further develop and implement new algorithms which can then transparently substitute those already present in the machine vision system.

7.1.1 Pixel intensity directed feature extraction

The first approach attempted to detect the first cutting river across a lace pattern and is based on the analysis of pixel intensity directed features in which various traditional image processing methods are used to extract the information from the scanned image.

By relying on this scheme, the extraction of the river heavily depends on the features of the repeated cutting path (refer to Section 4.3.1). As the lace web is deformed (distortion due to mechanism feed mis-alignments and/or changes in the pattern caused by the release of tension in the lace structure as it is cut), these features of the river are no longer presented. This causes detection of the river to fail (or even detection of the wrong path) when the construction or stretch in the lace web is bigger than 5-10 percent.

7.1.2 Fuzzy pattern recognition

In order to deal with the problems of material flexibility, a fuzzy reasoning rule-based technique is devised to analyse the distorted lace patterns (Section 4.3.2). According to numerous experimental results, the recognition system incorporating this technique can successfully extract the correct cutting paths within various lace motifs without prior knowledge of the lace patterns scanned. Up to 40 percent of lace distortion can be correctly analysed to obtain the cutting river. On most kind of lace motifs, about 300 milli-seconds (Intel 80486 DX2-66 processor) is required to extract the river using this algorithm.

However in a very few intricate lace patterns, such as shown in Figure 7-1, careful inspection shows that the river within the pattern is very tiny. After the thresholding operation and noise reduction (using average intensity analysis), a part of the river bank is wiped out, which is indicated in Figure 7-2. This will cause inaccuracy as this information is used to create the cutting path at the centre of the river banks.

To overcome this problem, the researcher has developed a scheme in which the program, firstly finds out the over-thresholding location within the pattern, then decreases a certain degree of threshold and re-bi-levels these problem areas of the image. To detect the locations of the river banks which might have caused the problem, the system measures the

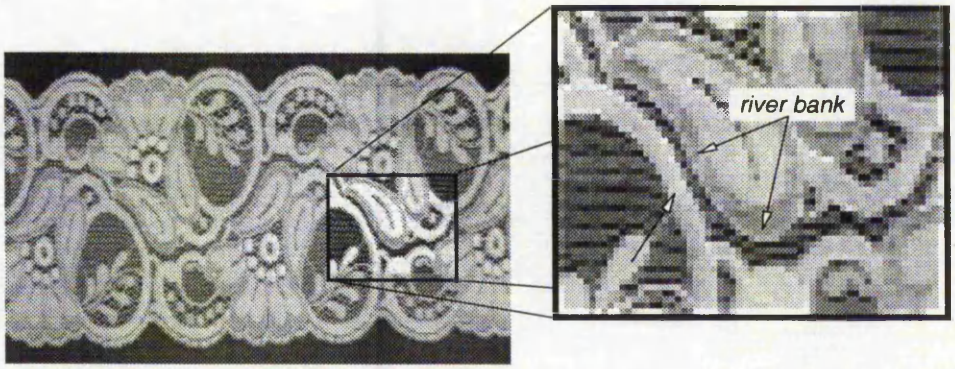


Figure 7-1: *Sample lace pattern with problem river bank*

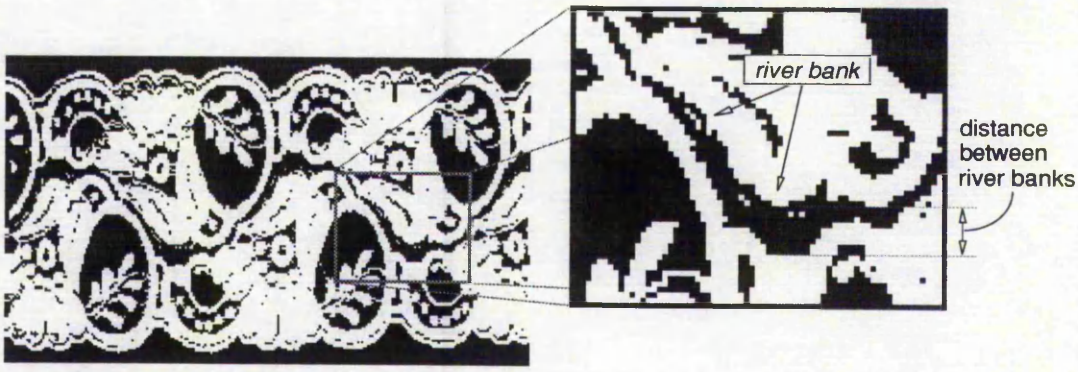
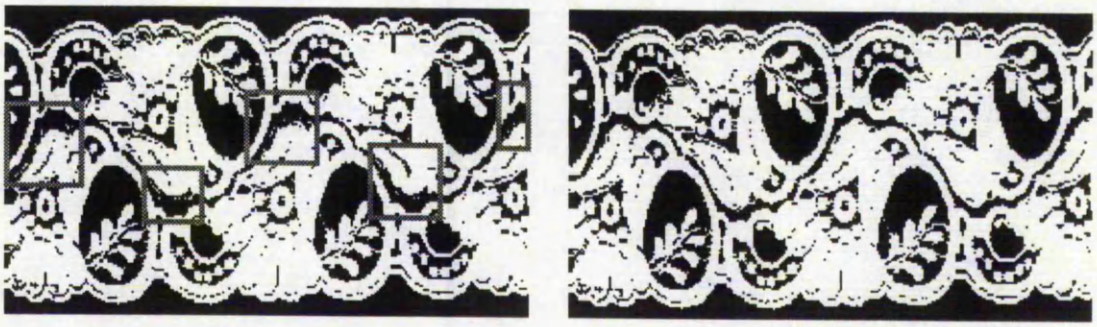


Figure 7-2: *After bi-leveling process, part of the river bank is wiped out*

distance between two river banks, which is indicated in Figure 7-2. If the distance is larger than a certain value, it can be this area conducted that may be over-thresholding. The threshold value is determined imprecisely. As the system inspects all the river banks over the entire frame, the problem areas are located (marked by squares in Figure 7-3 (a)).

Furthermore, the system decreases the thresholding point and uses it to bi-level these detected blocks within the gray-scaled image. These re-bi-leveled blocks are replaced into the original binary image in which the new bitmap is then used to create the cutting path. In contrast to the river constructed without the re-thresholding process, this scheme shows to have dramatic improvements. The locations of these problem blocks are recorded in a map which is used as reference for the bi-leveling operations in the subsequent frames. In other



(a) find out the problem areas (b) after re-thresholding process

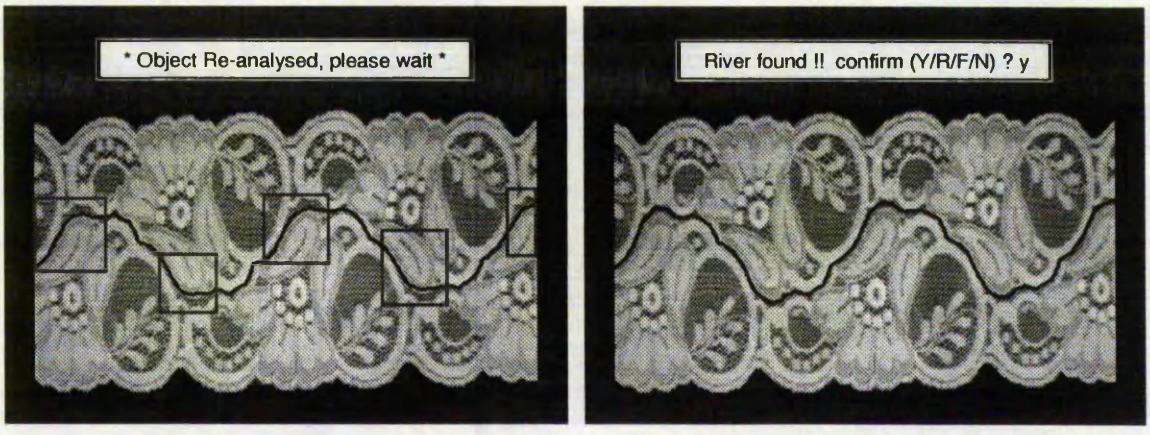
Figure 7-3: *Processes of fixing the problem areas*

words, as the system notices this difficulty, this knowledge can be continuously used for the following analysis. There is no need to carry out the double thresholding processes after the detection of the first frame.

However, since most of the lace motifs used do not involve such problem, in the current version of the programmes developed by the researcher, this function is set to a manual mode. As an engineer finds an unsatisfactory detection of the river, simply pressing *'R'* re thresholding will be initiated. The system will automatically detect the problem areas, re-threshold these blocks and create the new cutting path for the user, such as depicted in Figure 7-4.

7.1.3 Line mapping method

As the first cutting river is extracted from a lace pattern, this data is used as a reference path for the line mapping process (Section 4.4). According to numerous experimental results, the method shows to have fast response and high reliability. The speed for tracking the lace motif employing this method, using an Intel 80486 DX2-66 processor, is approximately 25 to 35 meters per minute. Higher performance could be obtained with a



(a) asked for re-thresholding (b) new detected cutting river

Figure 7-4: *Re-analysing the image to obtain a more accurate river*

better camera and frame grabber (faster rate of digitising a incoming video signal) or/and a faster host system, e.g., Pentium processor, DSP or Alpha CPU, etc.

7.1.4 Supervision of the system

As mentioned in Section 4.5.2, since the recognition system takes approximately two hundred milli-seconds to extract a cutting path from a lace pattern, this task will interrupt the cutting process between two captured frames. To solve the problem, we add a quarter of the repeat cutting cycle to the cutting river. Thus, while the machine is trimming past the *capture point* (see Figure 4-31), the vision system grabs a frame of lace image as well as finding the cutting path and transmitting the data to the cutting mechanism before the machine actually ends trimming. This enables continuous operation of the system in real-time.

Unfortunately, the controller in the selected testing rig is not a real-time environment - in the project we use a CNC machine as our cutting mechanism. The controller (68k processor) reads a batch of machine movement data from the host system and transforms them into a series of control commands to guide the motors leading the axes. This data transformation process is normally taken the 68k micro-controller more than 3 seconds to complete it. The non-real-time nature of the controller results in compromising the overall system throughput and not being able to precisely set the event priorities within the system to appropriate levels. Nevertheless, the selected cutting mechanism provides a satisfactory means of developing a prototype which is, from the cost and performance point of view, effective for experimentation.

7.1.5 Colour curves differentiation

In the post-processing vision system, a black and white camera with a monochrome frame grabber is employed to distinguish two different colour lines (Section 5.7.3.1). A

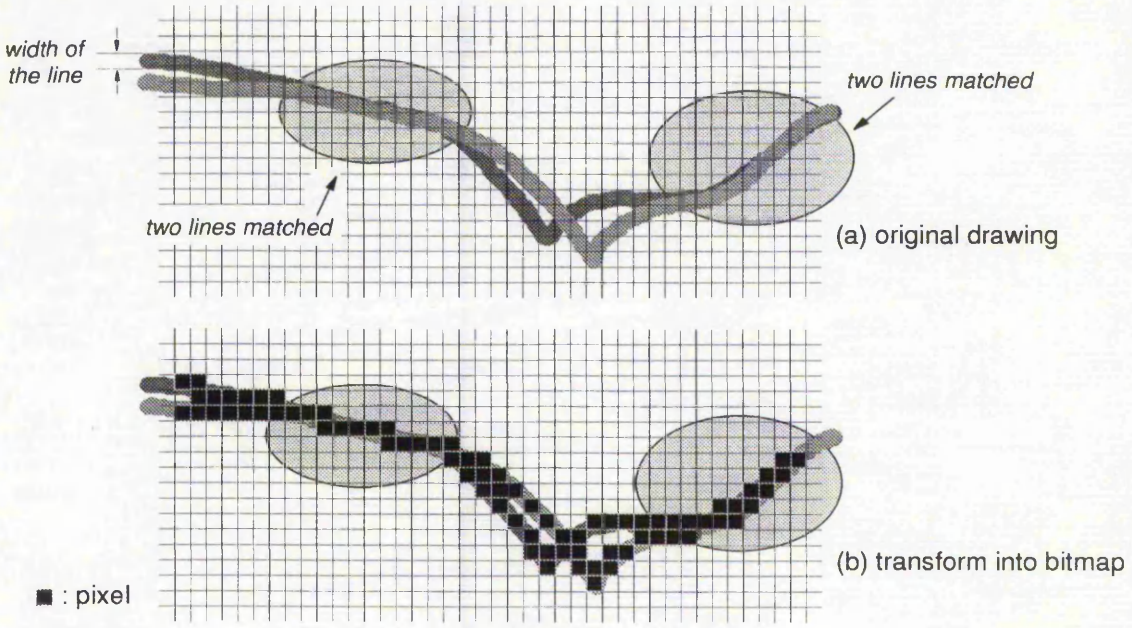


Figure 7-5: *Example of transforming two lines into bitmap*

colour curve differentiating technique has been developed to perform this task. According to various experiments, a number of difficulties have been found.

The resolution of the vision system is approximately 1 mm per pixel in which the width of the line drawn by the felt-tip pen is about 1 to 1.5 mm depending on the speed of drawing and the pressure applied to the pen. As depicted in Figure 7-5, the line captured by the camera is appeared between two grids of the pixel array. After the bi-leveling operation, part of the single line is recorded as two-pixel width (see Figure 7-5 (b)). This will cause inaccurate detection of the SMP following process (Section 5.7.3.3.1, fuzzy amplitude prediction) - If two captured lines are very close to each other, it is very difficult to detect the actual distance (path-following-error) between them. For example, in Figure 7-5 two lines are partly matched, these overlapped segments are detected either as one-pixel wide, or as two-pixels wide (marked by circles).

Of course, it is possible to develop a complex algorithm to 'understand' all these situations in order to conquer the problem described above. However this is not the main

objective in the project. Although the limitation of applying the colour curve differentiating technique to distinguish two very close lines has been found, this approach still works satisfactorily in other situation. The remaining problem will cause unreliable prediction for generating the *amplitude of the correction* while the scanned path is close or overlapped with the drawing path.

Since for cost effectiveness unsuitable low-resolution monochrome vision equipments are used, extra image analysing operations have to be undertaken in order to fulfill the objectives. These extra processes, such as line skeleton and curve recovery operations, will cause the detected (extracted) pattern to lose its original shape. In other words, the extracted pattern is only 'similar' to the original object, not exactly the same. The above mentioned problems will strongly affect the processes of correcting the path-following-errors. Since inexact algorithms are employed in the project, the tolerant nature of the techniques developed can be utilised to conquer and eliminate such difficulties. However, in this case, more learning steps are required for deriving the proper knowledge to entirely correct these errors.

It should be noted that the author strongly recommends to choose a suitable vision equipment rather than using the software functions to compensate the flaw caused by the inappropriate use of the sensing system. For instance, it is recommended to apply a colour camera and frame grabber instead of using a black/white camera in the post-processing vision system.

7.1.6 Three-vector method

This method has been devised to correct the SMP following problems. The algorithm (Section 5.7.3.2) is based on analysing the angles between three consecutive coordinates of the scanned path. The correction made by the analyser is only added in the Y coordinate of the scanned path. According to various experimental results, this method can deal with

irregular shapes of paths. Approximately 60 to 80 percent of path-following-error caused by the spring can be successfully removed.

If the correction made in a segment of the path is inaccurate, the rest of the segments of the path are also affected. Besides, the correction is only considered in the Y direction of the path. No matter how big the correcting magnitude given to the original path, when the curve in the desired scanned path is sharp (the angle of the curve is changed dramatically), the path-following-error cannot be entirely eliminated. From the experiences of investigating the 3VMethod, two important conclusions have been derived (refer to Section 5.7.3.2):

1) the correction of the SMP error has to be made in both X and Y coordinates, not only in one direction; and
2) it is better that the correction made for each of the segments in the path is self-reliant.

An improved algorithm based on this concept was developed. The algorithm produced excellent results which are discussed next.

7.1.7 Two-vector method

This method manipulates every two consecutive coordinates over the entire scanned path. As a line is connected between these two data points, the angles between the line and the X and Y coordinates are used to calculate the correcting energies for this segment (see Section 5.7.3.3). Inexact algorithms have been employed in the system to determine the correcting energies. Almost all path-following-errors between the paths can be eliminated. Three novel techniques based on the inexact algorithms have been applied to design and construct the inference kernel:

- fuzzy logic;
- neural fuzzy algorithm;
- and neural networks.

According to the experimental results, all these approaches have been shown to have the ability to successfully remove the errors created due to spring flexibility. Employing the fuzzy logic to design the inference engines (Section 5.7.3.3.1) has had several advantages including simplicity and ease of design. However, as the system complexity increases, it becomes difficult to determine the right set of rules and membership functions to describe the system behaviour. A significant amount of time is needed to properly tune the membership functions and adjust rules before a solution is obtained.

Artificial neural networks together with fuzzy logic were used (*neural fuzzy technique*, Section 5.7.3.3.2) to learn the system behaviour to solve the above mentioned problems. Using the system's input-output data, neural networks can learn the behaviour of the system and can accordingly generate fuzzy rules and membership functions. By proper learning, neural nets can develop good generalsiation capabilities and thus, can solve the problems of designing a complex system. The main idea in integrating the fuzzy logic controller with neural networks is to use the strength of each one collectively in the resulting neural fuzzy inference system. The fuzzy engine learns to adjust its performance automatically using the neural network structure and hence learns by accumulating experience.

The third approach, used to design the system, is to let a fuzzy controller supply the training data for a back-propagation neural network (Section 5.7.3.3.3) and use the trained network to determine an appropriate action to correct the error. The intention of utilising such a technique is to demonstrate that the large effort required for developing a high speed real-time fuzzy controller, which is generally achieved by adapting a fuzzy logic chip, can be avoided by designing a software fuzzy controller in a "quick-and-dirty" manner and combining it with a simple method to implement neural engine.

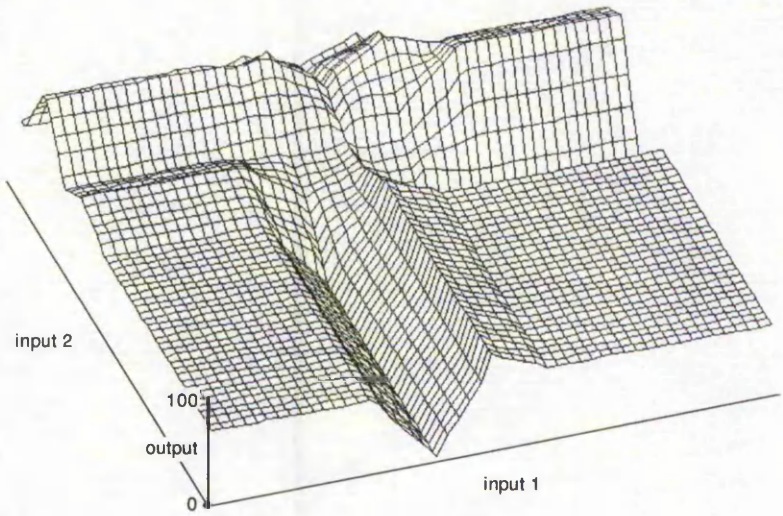


Figure 7-6: *The output pattern created by a 2-input / 1-output fuzzy engine*

The advantage of applying such an approach is that less recalling time is required by using the trained neural network kernel than the fuzzy controller. The larger the number of fuzzy rules, the longer is the time required for reasoning. Particularly when a considerable number of rules, e.g., more than 100 rules, are processed in the inference kernel of a fuzzy system, a significant amount of time can be reduced by using the neural network which learns the responses from the fuzzy system [BAL93][FRE93].

Figure 7-6 shows an example of applying a two-input / one-output fuzzy system to produce the learning pattern for training a standard fully connected three-layered neural network comprising two input neurodes, eight hidden neurodes and one output neurode. The network is trained with supervised back-propagation algorithm on the input / output data pairs created by the fuzzy kernel until it has learned the output actions determined by the fuzzy system. The trained neural engine basically becomes a clone of the fuzzy controller in the sense that its output behaviour (see Figure 7-7) imitates that of the fuzzy kernel.

It should be mentioned that a special scheme for fast convergence of a neural network has been devised to train the neural engines in the project. As the initial weights of a neural

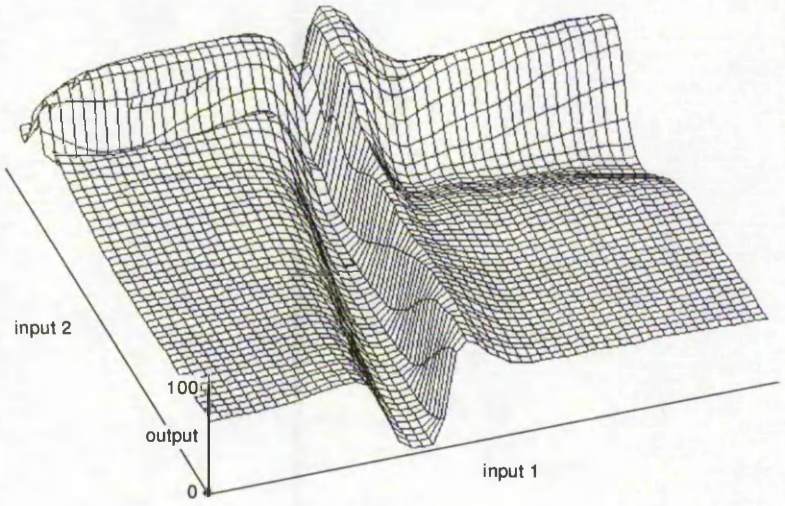


Figure 7-7: *The output pattern produced by the trained neural kernel*

net have been randomly assigned, the learning coefficient (η) and the momentum factor (α) (set to very high values - 0.9 and 0.95, respectively) are usually chosen for the problem in hand. This causes an unstable convergence of the neural net, because of the oscillation of the training process. The weights of the network connections are recorded while the smallest mean square error is detected. After several hundred learning iterations the last recorded connection weights, which have the smallest learning error, are used as a place to start another training process. Then an ordinary training procedure is applied. By using such an approach most of the *local energy minima* can be avoided while converging a neural network. This method has been employed to train the neural engines used in the Generic Error Compensation Algorithm stated in Chapter 6.

There are only twenty-four rules that have been derived in the fuzzy controller to express the operator's control actions in this project. Almost the same speed of reasoning process has been found in both fuzzy and neural kernels. In short, supposing that the results obtained in our experiments also hold in other environments, it might be reasonable to assume that a well-designed fuzzy controller employed as the teacher of a neural network will lead to further performance improvements in a combined fuzzy and neural approach.

7.1.8 Piecewise error compensation algorithm

The Piecewise Error Compensation Algorithm is derived to detect the correcting pattern and the correcting amplitude which are utilised by the 2VMethod for determining a compensated pattern. This algorithm is implemented by means of fuzzy logic (Section 5.7.3.3.1), neural networks (Section 5.7.3.3.2) and neural fuzzy technique (Section 5.7.3.3.3), respectively.

Through an on-line self-learning process, the intelligent controller can make an appropriate compensation to eliminate the deviation. According to the experimental results show that approximately three frames of correcting process are demanded to remove the errors.

7.1.9 Generic error compensation algorithm

In order to further improve the performance of correcting the SMP following problems, a *Generic Error Compensation (GEC) Algorithm* is developed and tested. After following a pre-defined template (square wave), the intelligent machine console analyses the result of the process. A neural network based kernel is engaged to learn to correct the deviation from this processed frame. A modified version of 2VMethod is applied to create the compensated path which is fed back to the controller. Applying the algorithm developed, only one frame of learning process is required to successfully remove the errors between the intended path and the actual path. Any regular and irregular shapes of paths can all be dealt with.

Since the A.I. engine learns the deviation pattern directly from the result of template following process, in contrast to the PEC Algorithm, the *GEC Algorithm* should produce more accurate outcomes. Of course the limitations of the vision system should be taken into account. The inaccuracy of the vision system limits the correcting capability of the A.I. kernel. However, if a sensing system can precisely detect the deviations between the desired

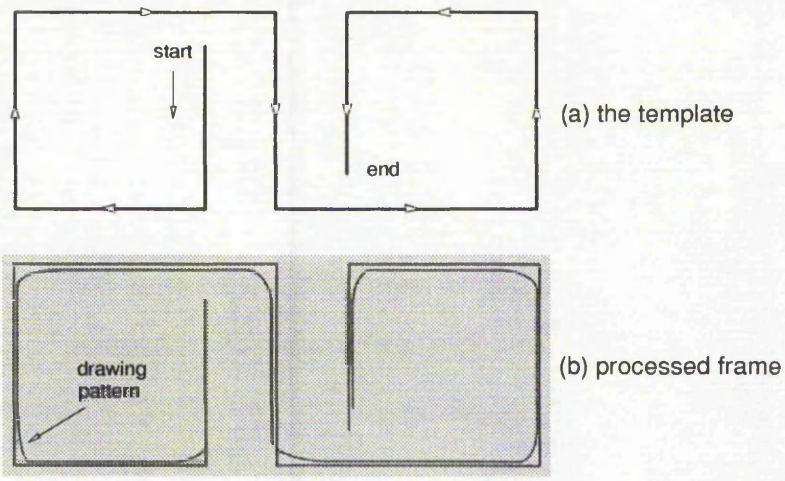


Figure 7-8: *(a) The new type of template;*

(b) A possible outcome of SMP following process

and the actual patterns. By utilising this approach, it is possible to remove all the errors due to flexibility of dynamic structures.

Since a conveyor system is attached within the test rig (as stated in Section 3.2), the cutting mechanism employed in the project is only controlled to move from +X to −X direction and +Y and −Y in both directions. However, in a general type of CNC machine the system normally can move from both +X to −X and −X to +X directions. Two dimensional patterns (−X/+X and −Y/+Y) can be drawn in this type of machine. In order to correct the true 2-D patterns, instead of following a square wave a new type of template has to be applied. Figure 7-8 represents this template as well as a possible outcome from following the template using the SMP. Four sets of correcting patterns can be derived by analysing this drawing.

Applying the same methods described in Sections 6.2 and 6.3, the deviation patterns and the maximum deviations (correcting amplitudes) can be detected. Figure 7-9 depicts this process. Extending the technique developed in the 2VMethod, the A.I. kernel can learn from the obtained deviation data and produces the compensated pattern. Figure 7-10 illustrates an example of an intended pattern and its derived compensated pattern using this scheme.

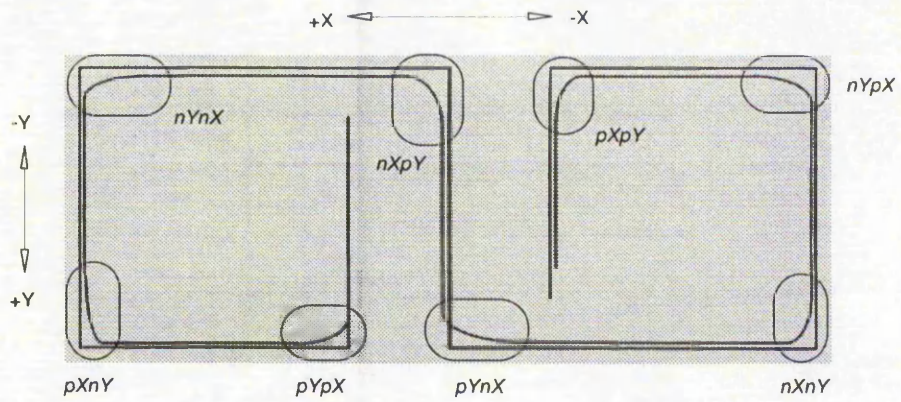


Figure 7-9: *Detecting the deviation patterns and the correcting amplitudes*

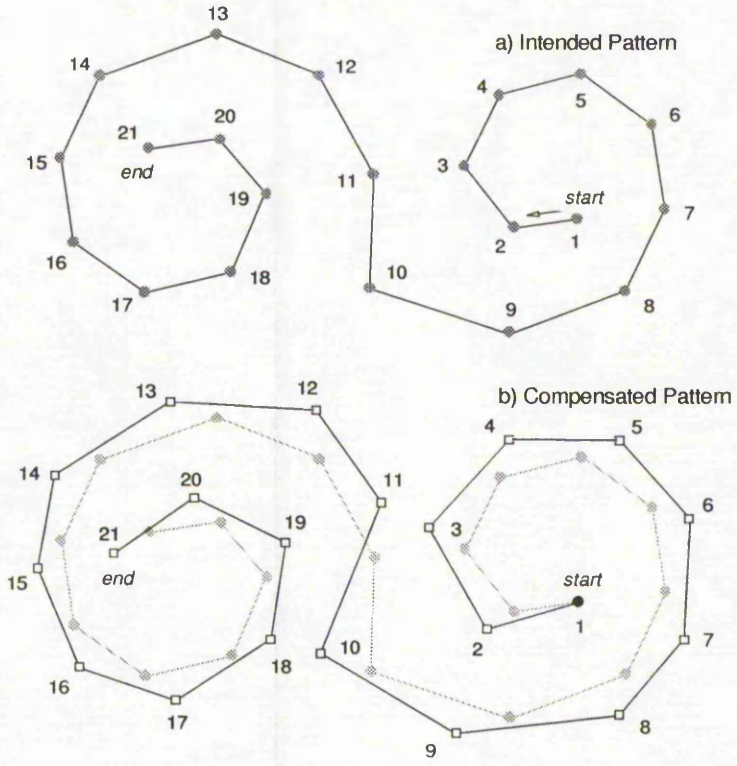


Figure 7-10: *Computing the compensated pattern from a desired pattern*

7.2 Future Work

This Section outlines the potential research areas in which the developed algorithms can be applied to solve problems where the characteristics of the system are complicated and difficult to mathematically model. Future research could entail compensation of vibrational

errors and/or control induced errors using inexact algorithms. Besides, the use of a well combined fuzzy logic / neural network controller could lead to improving the performance of automated manufacturing processes.

7.2.1 Compensation of vibrational errors and control induced errors

Manufacturing with high accuracy is influenced by numerous factors. These can be classified as follows: machine tool and its controlling equipment, workpieces, fixtures/jigs and tools. The listing must be completed by taking to consideration the environmental conditions [SZA93].

It is possible to propose a system in which sensors are used to collect the displacement data that indicates the movement of a end effector. This information is then fed into a intelligent controller which is based on the inexact algorithms developed. By comparing the *intended paths* and the *actual paths*, the controller can automatically learn to create the *compensation paths* for correcting the errors due to vibrational errors and/or control induced errors. This approach is essentially trying to avoid using very complex sensors to monitor all the system and environment factors, such as mentioned previously. Through comparing the difference between the required shape and the resultant shape, the controller can make the appropriate compensation (Figure 7-11).

As an illustration, a flat bed CNC cutting machine can be fitted with optical encoders which are used to monitor the actual movement of the end effector. A host controller can be used to produce movement commands to guide the machine tool. Due to the environmental influences (e.g., changing of temperature, dust content) and/or the original defects in the cutting machine (e.g., link flexibility, backlash, vibration), errors could be found between the intended output (from the controller) and the actual output. The neural fuzzy kernel is then employed here to *learn* the deviations (errors). This enables the system to make suitable compensation to correct the errors.

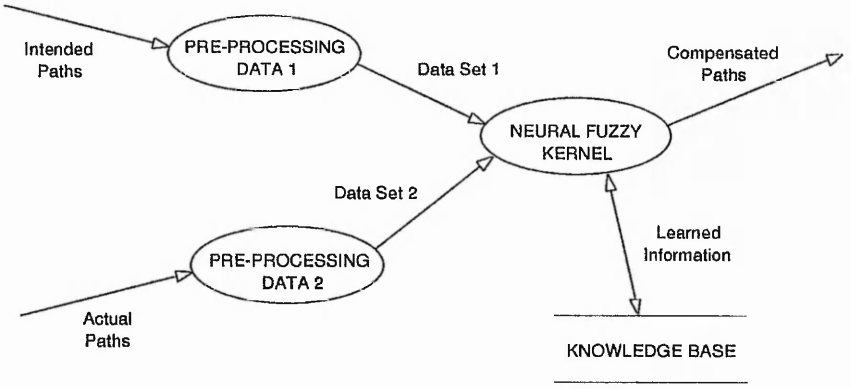


Figure 7-11: *Using the intelligent kernel to make appropriate compensation*

It is worth noting that although the optical encoders are proposed in the example system, different sensing system, such as infrared array sensors, high resolution CCD camera, etc., may also be considered too. In fact, as long as a sensing system can accurately detect the actual movement of the end effector, it will suffice.

7.2.2 Well combined fuzzy logic / neural network controller

As stated in Section 5.3.2, both fuzzy logic and neural networks are powerful design techniques which both have their strengths and weaknesses. Neural networks can learn from data sets, while fuzzy logic solutions are easy to deal with inexact information. A effective combination of the two technologies can deliver the best of both worlds. The Neural Fuzzy kernel was developed in the project to integrate the fuzzy logic controllers with neural networks. In addition, it is also possible to propose a system applying fuzzy logic and neural network engines respectively in terms of using the strength of each one collectively to solve very complex problems.

As illustrated in Figure 7-12, a neural engine reads a set of normalised data from the external sensing system. Since the neural networks is good for grouping massive data, it can be employed here to transform the data set into a linguistic term, which is then fed into a

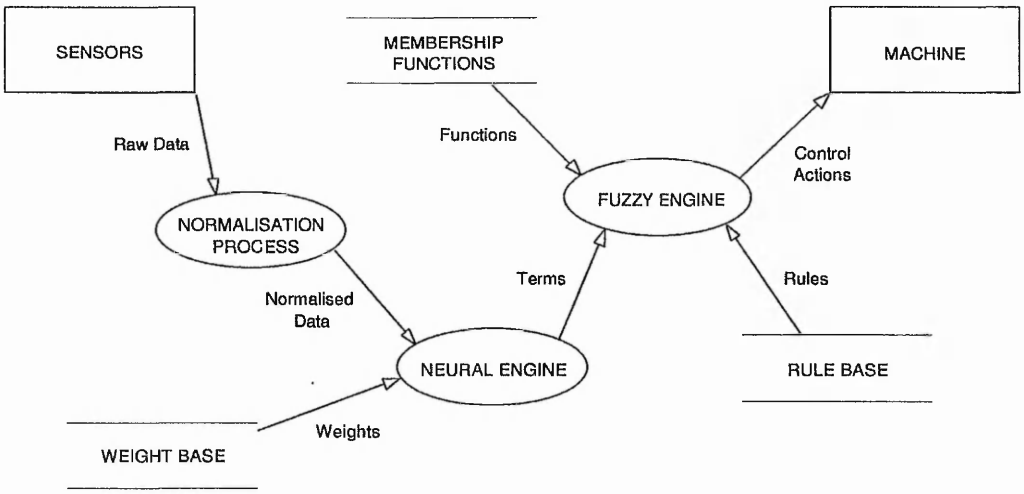


Figure 7-12: *Well-combined fuzzy logic / neural network kernel*

fuzzy engine. By utilising the rule base and the membership functions, the fuzzy engine can read the linguistic terms (antecedent) and generate a set of control actions to operate a machine.

The key benefit of fuzzy logic is that it allows description of the desired system behaviour using simple linguistic terms. This gives an engineer a simpler solution in less design time. However, in some cases, the knowledge that describes the desired system behaviour is contained in data sets, e.g., from the sensors. Here, the designer has to derive the fuzzy rule base from the data sets manually, which implies a major effort with large data sets.

To solve the above mentioned problem, a neural network can be proposed in the system for grouping a large data set into a fuzzy linguistic term. For example, an automated manufacturing system contains three sensors which continuously monitor the system and environmental factors: 1) speed of the monitor, 2) position of the tool and 3) temperature of the workpiece. The neural engine classifies the data sets from the sensing system into different groups such as *Speed is group* A_1, *Position is in group* B_1 and *Temperature is group* C_1, where group A, B and C are linguistic terms (fuzzy sets). These classified data

sets (terms) are fed into the fuzzy engine. Using the calculus of fuzzy if-then rules, these input terms are inferred and composed into a set of output terms. The defuzzification process takes place to produce the output value. For instance,

IF (S *is* A_1 and P *is* B_1 and T *is* C_1) THEN (CA *is* D_1)

IF (S *is* A_2 and P *is* B_3 and T *is* C_1) THEN (CA *is* D_2)

IF (S *is* A_2 and P *is* B_2 and T *is* C_3) THEN (CA *is* D_3)

: : : :

IF (S *is* A_i and P *is* B_j and T *is* C_k) THEN (CA *is* D_n)

where S is Speed, P is Position, T is Temperature and CA is Control Action. As the A_i , B_j and C_k are obtained from the neural engine, these sets are used by the fuzzy engine to produce a set of Control Actions (D_n) for controlling a machine.

Synthesis of fuzzy logic and neural networks offers a key advantage over traditional control systems. It provides model-free estimation of a control system. The user need not specify how the controller's output mathematically depends on its input data. The user only needs to provide a statistically representative set of numerical training samples of the system (e.g., the sensing data sets) and structured knowledge of the control process for developing the intelligent kernel. It is believed that a well combined fuzzy logic / neural networks controller, such as described above, can lead to an improved design and implementation of a control system as well as one that is more effective to maintain.

8. CONCLUSIONS

Chapter 8
CONCLUSIONS

A vision based intelligent machine control system has been developed. This system uses automated lace trimming as a suitable platform for experimentation and assessment of the effectiveness of using remote sensing for control and monitoring. In order to monitor the processed objects as well as using the fed-back information to correct the processing errors, a closely integrated remote sensing based control using the pre- and post-processing vision stations has been designed and implemented.

The objective of detecting the river in an unseen lace pattern in real-time has been achieved. This has enabled the development of a working prototype for an automatic lace scalloping machine. As the complexity of lace patterns, it is very difficult to detect the cutting paths within various lace patterns with no previous knowledge. Besides, it is found that the biggest problem in automating the process of lace scalloping is that of dealing with lace distortion in real time. Distortion, not only creates problems for the pattern recognition task, it also complicates the feeding and cutting processes.

Comparing the two approaches used in traditional image processing [SHE94a] and the fuzzy pattern recognition technique to find the first river without prior knowledge, the fuzzy logic based approach is more effective. Using the fuzzy technique, the river within the lace pattern with up to 40% contraction can be successfully detected. However, the traditional method has failed under the same circumstance. According to the experimental results, a combination of fuzzy pattern recognition technique and the line mapping method can be applied to detect the distorted river within various lace patterns in real time. Besides, in contrast to the schemes mentioned in [RUS88] and [KIN93], the proposed algorithm is not only easier to design and implement, it is also more effective in coping with distortion.

Furthermore, the presented method does not require any training or prior knowledge of the lace pattern.

An innovative approach based on modeling human operators' experience and control actions using *inexact algorithms* are developed to solve the complex problem of material flexibility. The experimental system uses the pre- and post-processing vision station to closely couple the feedback information to the controller. By translating a skilled operator's knowledge into a set of linguistic terms or groups of network connections, the intelligent machine can learn from experience (on-line learning) and self-adjust the control actions to match the desired objective. A.I. engines are constructed in order to determine suitable actions to correct the processing errors.

A spring mounted pen is used in the experiments to emulate the movement of the lace strip due to the cutting forces caused by tactile cutting and the transportation mechanism. This system has been implemented in terms of fuzzy logic, neural networks and neural fuzzy technique respectively. Numerous experiments have been carried out to evaluate the effectiveness of this approach. According to various experiments, applying the Piecewise Error Compensation Algorithm, requires approximately three frames of training process before the machine reacts correctly in minimising the error. Furthermore, only one frame of training process is necessary by means of utilising the Generic Error Compensation Algorithm to compensate for the errors. Both learning techniques can successfully deal with any regular and irregular shape of paths, and can produce excellent outcome even better than a human operator.

The development of the system is a novel approach to flexible material processing and has further applications where modeling system behaviour characteristics is difficult. Examples of such systems can range from controlling a robot moving on a slippery surface, driving a car on snow or piloting a boat, etc. Furthermore, by relying on the intelligent software kernel together with the vision system the controller no longer needs to rely on

accurate position fed-back from sensors. Backlash, joint flexibility, poor feedback and stick slip [REA91][STE90] can potentially be compensated for by the controller. While the characteristics of the mechanism, such as component wear, temperature variation, change over time, the controller can automatically make appropriate compensation. It is sensible to anticipate that computer hardware will decrease continuously in cost while increasing in performance. In contrast, mechanical hardware costs are more likely to stay in line with inflation in the future years [HOD95]. Consequently, it is reasonable to make a shift from mechanical hardware to computer with the associated intelligent software kernel in automated industrial applications.

REFERENCES

References

[ACK94]

Ackermann, J.; Guldner, J. and Utkin, V.I. "A robust nonlinear control approach to automatic path tracking of a car", IEE Conference Publication. No.389 - Control '94, pp.196-201, 21-24 March 1994.

[ALK90]

Aleksander, Igor and Morton, Helen. An Introduction to Neural Computing, Chapman and Hall, London, 1990.

[ALT94]

Altrock, Constantin von. "NeuroFuzzy technologies", Sensor Review, vol.14, No.3, pp.15-17, 1994.

[ANG90]

Angell, Ian O. High-resolution Computer Graphics Using C, Macmillan Education Ltd., London, 1990.

[BAL93]

Balazinski, M.; Czogala, E. and Sadowski, T. "Control of metal-cutting process using neural fuzzy controller", IEEE Proceedings, 0-7803-0614-7, pp.161-166, 1993.

[BAN94]

Bannatyne, R. "Development of fuzzy logic in embedded control", Sensor Review, Vol.14, No.3, pp.11-14, 1994.

[BER88]

Bernard, John A. "Use of a rule-based system for process control", IEEE Proceedings, 0272-1708/88/10000-0003, pp.3-13, October 1988.

[BER90]

Berenji, Hamid. "Neural networks and fuzzy logic in intelligent control", TH0333-5 IEEE Proceedings, pp.916-920, 1990.

[BEZ81]

Bezdek, James C. Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, London, 1981.

[BLA89]

Blackwell, George F. "Machine vision in the tire industry", IEEE Proceedings, CH2764-9/89/0000-0067, pp.67-79, 1989.

[BRI92]

British Patent Application 9216643.8 "Automatic Operations on Materials", Filed 5 August 1992.

[BUC94]

Buckley, james J. and Hayashi, Yoichi. "Fuzzy neural networks", Fuzzy Sets. Neural Networks. and Soft Computing, Van Nostrand Reinhold, pp.233-249, 1994.

[BUS83]

"Business opportunities in advanced technology - robotics", Government of New South Wales Department of Industrial Development and Decentralisation, Sydney, 1983.

[CEL92]

Cela, A.; Hamam, Y. and Zhang, H. "Towards a neural fuzzy controller", IEEE Proceedings, 0-7803-0720-8, pp.1277-1282, 1992.

[CHE93]

Chen, C H; Pau, L F and Wang, P S P. Handbook of pattern Recognition & Computer Vision, Word Scientific Publishing, London, 1993.

[CHO93a]

Chow, Mo-yuen and Goode, Paul V. "Adaptation of a neural/fuzzy fault detection system", Proceedings of the 32nd Conference on Decision and Control. IEEE, pp.1733-1738, 1993.

[CHO93b]

Chow, Mo-yuen; Sharpe, Robert N. and Hung, James C. "On the application and design consideration of artificial neural network fault detectors", IEEE Transactions on Industrial Electronics, Vol.40, No.2, pp.181-198, April 1993.

[CHO93c]

Chow, Mo-yuen; Thrower, James P. and Taylor, Leroy S. "Neural-fuzzy hybrid system for distribution fault causes identification", IEEE Proceedings, 0-7803-1217-1/93, pp.427-431, 1993.

[COH92]

Cohen, M. E. and Hudson, D. L. "Approaches to the handling of fuzzy input data in neural networks", IEEE Proceedings, 0-7803-0236-2/92, pp.93-100, 1992.

[CON85]

Conte, Gianni and Del Corso, Dante. Multi-microprocessor Systems for Real-Time Applications, D. Reidel Pub. Co., 1985.

[CZO82]

Czogala, E. and Pedrycz, W. "Fuzzy rules generation for fuzzy control", Cybernetics and Systems: An International Journal, pp.275-293, 1982.

[DAY90]

Dayhoff, Judith E. Neural Network Architectures : An Introduction, Van Nostrand Reinhold, new York, 1990.

[DEV85]

Devi, B. B. and Sarma V. V. S. "Estimation of fuzzy memberships from histograms", Information Sciences 35, Elsevier Science Publishing Co., Inc., pp.43-59, 1985.

[DOY95]

Doyle, Rory. "Neurofuzzy modelling and control", Electro Technology, pp.10-12, April/May 1995.

[EBE90]

Eberhart, Russell C. and Dobbins, Roy W. Neural Network PC Tools - A Practical Guide, Academic Press, Inc., London, 1990.

[EDM91]

Edmonds, J.M. and Davies, E.R. Microprocessors and Microsystems 15, No. 1, pp.11-19, 1991.

[EFS87]

Efstathiou, J. "Rule-based process control using fuzzy logic", Approximate Reasoning in Intelligent Systems, Decision and Control, Pergamon Press, Oxford, pp.145-158, 1987.

[EKE92]

Ekerol, H. and Hodgson, D.C. Mechatronics 2, pp.555-565, 1992.

[FAI88]

Fairhurst, Michael C. Computer Vision for Robotic Systems - An Introduction, Prentice Hall, London, 1988.

[FER90]

Ferraro, Richard F. Programmer's Guide to the EGA and VGA Cards - Second Edition, Addison-Wesley Publishing Company, Inc., Wokingham, England, 1990.

[FIS87]

Fisher, M A; Firshan, O. Readings in Computer Vision, Morgan Kaufman, 1987.

[FRE88]

Freeman, Herbert. Machine Vision - Algorithms, Architectures, and Systems, Academic Press, Inc., London, 1988.

[FRE93]

Freisleben, Bernd and Kunkelmann, Thomas. "Combining fuzzy logic and neural networks to control an autonomous vehicle", IEEE Proceedings, 0-7803-0614-7/93, pp.321-326, 1993.

[FUK92]

Fukuda, Toshio; Shimojima, Koji; Arai, Fumihito and Matsuura, Hideo. "Multi-sensor integration system based on fuzzy inference and neural network for industrial application", IEEE Proceedings, 0-7803-0236-2, pp.907-914, 1992.

[GAL90]

Galbiati, Louis J. Machine Vision and Digital Image Processing Fundamentals, Prentice Hall, London, 1990.

[GON87]

Gonzalez, Rafael C. Digital Image Processing : Second Edition, Addison-Wesley Publishing Company, England Amsterdam, 1987.

[HAR92]

Haralick, Robert M.; Shapiro, Linda G. Computer and Robot Vision -Volume I, Addison-Wesley Publishing Company, Wokingham, England, 1992.

[HAR93]

Haralick, Robert M.; Shapiro, Linda G. Computer and Robot Vision -Volume II, Addison-Wesley Publishing Company, Wokingham, England, 1993.

[HIR89]

Hirota, Kaoru; Arai, Yoshinori and Hachisu, Shiroh. "Fuzzy controlled robot arm playing two-dimensional ping-pong game", Fuzzy Sets and Systems 32, pp.149-159, 1989.

[HIR92]

Hirota, Kaoru; KYO, Yokou; Ohtani, Masayuki and Yubazaki, Naoyoshi. "A target tracking robot based on fuzzy control", IEEE International Workshop on Robot and Human Communication, pp.335-340, 1992.

[HOD95]

Hodges, S.E. and Richards, R.J. "Looking for a better robot: visual robot control for cheap, flexible assembly", Image Processing and Its Applications, IEE Conference Publication No.410, pp.707-711, July 1995.

[HOL84]

Hollingum, Jack. Machine Vision - The Eyes of Automation, IFS (Publications) Ltd., U.K., 1984.

[HUA90]

Huang, Liang-Jong and Tomizuka, Masayoshi. "A self-paced fuzzy tracking controller for two-dimensional motion control", IEEE Transactions on Systems, Man, and Cybernetics, Vol.20, No.5, pp.1115-1124, Sep./Oct. 1990.

[ISA93]

Isaka, Satoru. "Fuzzy temperature controller and its application", SPIE, Vol.2061, pp.59-65, 1993.

[ISI87]

Isik, C. "Identification and fuzzy rule-based control of a mobile robot motion", IEEE Proceedings, TH178-4/87/0000/0094, pp.94-99, 1987.

[KAN79]

Kandel, A. and Lee, S. C. Fuzzy Switching and Automata: Theory and Applications. Crane, Russak & Company, Inc., London, 1979.

[KAN92]

Kandel, Abraham. Fuzzy Expert Systems. CRC Press, London, 1992.

[KAM90]

Kanada, H.; Gotoh, T.; Yoshida, M. "A visual control system using image processing and fuzzy inference", Transactions of The Institute of Electronics. Information and Communication Engineers D-II. Vol. J73d-II, Is 12, pp.1967-1975, Dec. 1990.

[KAU85]

Kaufmann, Arnold and Gupta, Madan M. Introduction to Fuzzy Arithmetic - Theory and Applications. Van Nostrand Reinhold Company, New York, 1985.

[KEM83]

Kemp, D.R. et. al. "A sensory gripper for handling textiles", 13th. Internal Symposium on Industrial Robots, Chicago, U.S.A., pp.17-21, April 1983.

[KHA93]

Khan, Emdad and Venkatapuram, Prahlad. "Neufuz: neural network based fuzzy logic design algorithms", IEEE Proceedings, 0-7803-0614-7 , pp.647-654, 1993.

[KIN93]

Knig, T.; Tao, L.G.; jackson, M.R. and Preston, M.E. "Real-time tracking of patterns on deformable materials using DSP", IEE International workshop on Systems Engineering for Real-Time Applications, Royal Agricultural College, Cirencester, U.K. pp.178-183, Sep. 1993.

[KLI88]

Klir, George J. and Folger Tina A. Fuzzy Sets. Uncertainty. and Information. Prentice Hall, New Jersey, 1988.

[KRI92]

Krishnapuram, Raghu and Keller, James M. "Fuzzy set theoretic approach to computer vision: an overview", IEEE Proceedings, -07803-0236-2/92, pp.135-142, 1992.

[LEE89]

Lee, D. G. Jr. "Preliminary results of applying neural networks to ship image recognition", Proc. Int'l. Joint Conf. on Neural Networks, Washington, D.C., II:576, 1989.

[LEE90]

Lee, C. C. "Fuzzy logic in control systems: fuzzy logic controller - Part I and II", IEEE Transactions on SMC, Vol.20, No.2, March/Avril 1990.

[LEE92]

Lee, C. S. George and Lin, C. T. "Supervised and unsupervised learning with fuzzy similarity for neural-network-based fuzzy logic control systems", IEEE Proceedings, 0-7803-0720-8/92, pp.688-693, 1992.

[LEF91]

Lefevre, P.; Neyer, M. De; Gorez, R. and Barreto, J. "Fuzzy internal models in vision systems modelling", IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS '91, pp.105-110, November 1991.

[LEN93]

Leng, Tay Sam; Leng, Nah Swee and Teck, Ng We. "Control of an inverted pendulum using a neural-fuzzy controller", IEEE Proceedings, 0-7803-1223-6/93, pp.212-217, 1993.

[LEW93]

Lewis, F.L.; Yesildirek, K. and Liu, K. "Neural net controller with guaranteed stability", IEEE Proceedings, 0-7803-1485-9, pp.103-108, 1993.

[LIN91]

Lin, C. T. and Lee, C. S. G. "Neural-network-based fuzzy logic control and decision system", IEEE Transactions on Computers, Vol.40, No.12, pp.1320-1336, Dec. 1991.

[LIN94]

Lin, Chin-Teng and Lee, C. S. George. "Reinforcement structure / parameter learning for neural-network-based fuzzy logic control systems", IEEE Transactions on Fuzzy Systems, Vol.2, No.1, pp.46-63, February 1994.

[LIND91]

Lindley, Craig A. Practical Image Processing in C, John Wiley & Sons, Inc., New York, 1991.

[LIP87]

Lippmann, Richard P. "An introduction to computing and neural nets", IEEE ASSP Magazine, pp.4-21, April 1987.

[LOW91]

Low, Adrian. Introductory Computer Vision and Image Processing, McGRAW-HILL Book Company, London, 1991.

[MAC92]

Machine Reference Manual. Pacer Systems Limited, Nottingham, United Kingdom, Revised April 1992.

[MAI85]

Maiers J. and Sherif Y. S. "Applications of fuzzy set theory", IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-15, No.1, pp.175-189, 1985.

[MAM74]

Mamdani, E.H. "Applications fo fuzzy algorithms for simple dynamic plant", Proc. IEE, Vol.121, No.12, pp.1585-1588, 1974.

[MAT85]

Matsushima, Kozo and Sugiyama, Hideyuki. "Human operator's fuzzy model in man-machine system with a nonlinear controlled object", Industrial Application of Fuzzy Control, pp.175-185, 1995.

[MCC91]

McCord, James W. Borland C++ Programmer's Guide to Graphics, SAMS, Indiana, 1991.

[MIL90]

Miller, Sutton and Werbos editors. Neural Networks for Control, The MIT Press, Cambridge, MA, 1990.

[MIN69]

Minsky, M and Papert, S. Perceptrons, MIT Press, 1969.

[MIZ94]

Mizumoto, M. "Multifold fuzzy reasoning as interpolative reasoning", Fuzzy Sets, Neural Networks, and Soft Computing, Van Nostrand Reinhold, New York, pp.188-193, 1994.

[NAR90]

Narendra, Kumpati S. and Parthasarathy, Kannan. "Identification and control of dynamical system using neural networks", IEEE Transaction on Neural Networks, Vol.1, No.1, March 1990.

[NEG85]

Negoita, Constantin Virgil. Expert Systems and Fuzzy Systems, The Benjamin / Cummings Publishing Company, Inc., London, 1985.

[NOM92]

Nomura, Hiroyoshi; Hayashi, Isao and Wakami, Noboru. "A learning method of fuzzy inference rules by descent method", IEEE Proceedings, 0-7803-0236-2/92, pp.203-210, 1992.

[NOR91]

Norton-Wayne, L. "Inspection of lace using machine vision" , Computer Graphics Forum, Vol. 10, Iss. 2, pp.113-119, June 1991.

[NOV89]

Novak, Vilem. Fuzzy Sets and Their Applications, Adam Hilger, England, 1989.

[OKAD92]

Okada, H.; Watanabe, N.; Kawamura, A. and Asakawa, K. "Initialising multilayer neural networks with fuzzy logic", Proceeding of the International Joint Conference on Neural Networks, Vol.1, pp.239-244, 1992.

[OKAW92]

Okawa, Yoshikuni and Aoki, Jun. "Fuzzy control of a mobil robot for the push-a-box operation", Procedure of the 1992 IEEE International Conference on Tools with AI, pp.172-179, November 1992.

[OLL91]

Ollero, A. and Amidi, O. "Predictive path tracking of mobile robots, application to the CMU NavLab", IEEE Proceedings, 7803-0078/91/0600-1081, pp.1081-1086, 1991.

[PAL92]

Pal, S. and Mitra, S. "Multilayer perceptron, fuzzy sets, and classification", IEEE Transactions on Neural Networks, Vol.3, No.5, pp.683-697, September 1992.

[PRO90]

Proceedings of the SPIE - The International Society for Optical Engineering. Intelligent Robots and computer Vision VIII: Algorithms and Techniques, Country of Publication, USA, Vol. 1192, Iss. pt.1, 1990.

[RAN92]

Ranganath, H.S.; Chipman, L.J. "Fuzzy relaxation approach for scene matching" , Image and Vision Computing, Vol. 10, Is 9, pp.631-640, Nov. 1992.

[REA91]

Readman, Mark and Belanger, Pierre. "Acceleration feedback for flexible joint robots", IEEE Proceedings of the 30th Conference on Decision and Control, England, pp.1385-1390, December 1991.

[ROV91]

Rovetta, Alberto and Wen, Xia. "Fuzzy logic in robot grasping control", IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS '91, pp.1632-1637, November 1991.

[RUM86]

Rumelhart, D.E. and McClelland, J.L. Parallel Distributed Processing. Explorations in the Microstructure of Cognition. Vols.1 and 2. MIT Press, Cambridge, MA, 1986.

[RUS88]

Russell, R. A. and Wong, P. "Automation of lace cutting using computer vision", Robots: Coming of Age. Proceedings of the International Symposium and Exposition on Robots. Designed the 19th ISIR by the International Federation of Robotics, pp.385-393, Nov. 1988.

[SAM91]

Sampei, M.; Tamura, T.; Itoh, T. and Nakamichi, M. "Path tracking control of trailer-like mobile robot", IEEE/RSJ International Workshop on Intelligent Robots and Systems, Cat. No. 91TH0375-6, pp.193-198, November 1991.

[SAN84]

Sanchez, E. Fuzzy Information. Knowledge Representation and Decision Analysis. Pergamon Press, Oxford, 1984.

[SAN87]

Sanchez, E. and Zadeh L. A. Approximate Reasoning in Intelligent Systems. Decision and Control, Pergamon Press, Oxford, 1987.

[SAN95]

Sanby, C. and Norton-Wayne L. "Machine vision inspection of lace using a neural network", SPIE Vol.2423, 0-8194-1770-X, pp.314-322, 1995.

[SAS88]

Sasaki, Tsuna and Akiyama, Takamasa. "Traffic control process of expressway by fuzzy logic", Fuzzy Sets and Systems 26, pp.165-178, 1988.

[SHA88]

Shao, Shihuang. "Fuzzy self-organizing controller and its application for dynamic processes", Fuzzy Sets and Systems 26, pp.151-164, 1988.

[SHE94a]

Sherkat, Nasser; Mike Birch, Peter Thomas. "Real-time vision for automatic lace cutting" , IS&T/SPIE Symposium on Electronic Imaging Science & Technology, California, USA, pp.322-333, 6-10 February 1994.

[SHE94b]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Applications, Kawasaki, Japan, pp.61-64, 13-15 December, 1994.

[SHE95a]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology, California, USA, pp.323-333, 5-10 February, 1995.

[SHE95b]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "Tightly coupled vision based control system using inexact algorithms", IEEE Second Asian Conference on Computer Vision, Singapore, 1995.

[SHE96a]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A Vision Based Control System Using Neural Fuzzy Theory", Fourth Iranian Conference on Electrical Engineering, ICEE-96, Tehran, Iran, 1996.

[SHI91]

Shin, Dong Hun; Singh, Sanjiv and Shi, Wenfan. "A partitioned control scheme for mobile robot path tracking", IEEE Proceedings, CH3051-0/91/0000-0338, pp.338-342, 1991.

[SHI95a]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE Fifth International Conference on Image Processing and Its Applications, Edinburgh, U.K., 1995.

[SHI95b]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Automation of lace cutting using real-time vision", 8th International Congress on Condition Monitoring and Diagnostic Engineering Management, Queen's University, Canada, 1995.

[SHI96a]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Close coupling of pre- and post-processing vision stations using Inexact Algorithm", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology, California, USA, 1996.

[SHI96b]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Correction of errors due to flexibility of dynamic systems", 1996 IEEE International Conference on Robotics and Automation, Minnesota, U.S.A., 1996.

[SHI96c]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "An automatic lace trimming process using real-time vision", Journal of Real-Time Imaging, 1996.

[SHI96d]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Inexact Algorithm for correction of errors due to flexibility of dynamic structures", IEEE 8th Mediterranean Electrotechnical Conference - Industrial Applications in Power Systems, Computer Science and Telecommunications, Bari, Italy, 1996.

[SHI96e]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Neural fuzzy based self-learning algorithms for handling flexibility of dynamic structures", The 22nd Annual International Conference of the IEEE Industrial Electronics Society, Taipei, Taiwan, 1996.

[SHI96f]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Generic error compensation algorithm for managing flexibility of dynamic structures using neural fuzzy approaches", 1996 IEEE International Conference on Systems, Man and Cybernetics, Beijing, China, 1996.

[SON89]

Song, Yong-duan; Gao, Wei-bing and Cheng, Mian. "Study of path tracking of robot manipulators with unknown payload", IEEE Proceedings, CH2767-2/89/0000-0321, pp.321-324, 1989.

[SON93]

Sonka, Milan, Hlavac, Vaclav and Boyle, Roger. Image Processing, Analysis and Machine Vision, Chapman & Hall Computing, London, 1993.

[STE90]

Stein, Jeffrey L. and Wu, Wei. "Monitoring stick-slip motion of machine tool slides using feed motor current", IEEE International Workshop on Intelligent Motion Control, pp.611-615, August 1990.

[STI92]

Stipanicev, Darko and Cecic, Mirjana. "Eye-hand coordination based on fuzzy vision transducer", IEEE Proceedings, 0-7803-0236-2/92, pp.2935, 1992.

[SUG85]

Sugeno, M. and Nishida, M. "Fuzzy control of model car", Fuzzy Sets and Systems 16, pp.103-113, 1985.

[SUH91]

Suh, Hong; Kim, Tae Won; Heu, Shin and Oh, Sang-Rok. "Visual servoing by a fuzzy reasoning method", IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS '91, pp.111-116, November 1991.

[SUN94]

Sun, Chuen-Tsai. "Rule-based structure identification in an adaptive-network-based fuzzy inference system", IEEE Transactions on Fuzzy Systems, Vol.2, No.1, pp.64-73, February 1994.

[SZA93]

Szabo, Otto. "Installation of an ultraprecisional CNC lathe laboratory, its accuracy and application", Mechatronics, Vol.3, No.2, pp.215-219, 1993.

[TAK83]

Takagi, T. and Sugeno, M. "Derivation of fuzzy control rules from human operator's control actions", IFAC Information Marseille, France, pp.55-60, 1983.

[TAN87]

Tang, Kwok L. and Mulholland, Robert J. "Comparing fuzzy logic with classical controller designs", IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-17, No.6, pp.1085-1087, 1987.

[TAY86]

Taylor, P.M. et. al. "Sensory robotic systems for the garment and shoe industries", Proc. 16th ISIR, Brussels, Belgium, 1986.

[VER91]

Vernon, David. Machine Vision - Automated Visual Inspection and Robot Vision, Prentice Hall, London, 1991.

[VIL90]

Villalobos, Leda and Gruber, Sheldon. "A system for neural network-based inspection of machined surfaces", Journal of Neural Network Computing, Auerback, Vol.2, No.2, pp.18-30, 1990.

[VIO93]

Viot, Greg. "Fuzzy logic in C: creating a fuzzy-based inference engine", Dr. Dobb's Journal, pp.40-49 & 94, February 1993.

[WEI91]

Weiskamp, Keith; Heiny, Loren. Power Graphics Using Turbo C++, John Wiley & Sons, Inc., New York, 1991.

[WHI92]

White, David A. and Sofge, Donald A. Handbook of Intelligent Control - Neural, Fuzzy, and Adaptive Approaches, Van Nostrand Reinhold, New York, 1992.

[WOL91]

Wolfe, D.F.H.; Wijesoma, S.W. and Richards R.J. "Eye-to-hand co-ordination", Assembly Automation. MCB University Press. Vol.11, No.1, pp.15-20, 1991.

[WON86]

Womg, P.C. "Robotics R&D in the garment industry", Automach Australia '86, Sydney, Australia, May 1986.

[XU87]

Xu, Chen-Wei and Lu, Yong-Zai. "Fuzzy model identification and self-learning for dynamic systems", IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-17, No.4, pp.683-689, 1987.

[YAG92a]

Yager, Ronald R. "Fuzzy sets and approximate reasoning in decision and control", IEEE Proceedings, 0-7803-0236-2, pp.415-428, 1992.

[YAG92b]

Yager, Ronald R. and Zadeh, Lotfi A. An Introduction to Fuzzy Logic Applications in Intelligent Systems, Kluwer Academic Publishers, London, 1992.

[YAN94]

Yanger, R.R. and Zadeh L.A. Fuzzy Sets, Neural Networks, and Soft Computing, Van Nostrand Reinhold, New York, 1994.

[YEN91]

Yen, John and Pfluger, Nathan. "Designing an adaptive path execution system", IEEE Proceedings, ISSN# 0-7803-0233-8/91, pp.1459-1464, 1991.

[YAS83]

Yasunobu, S.; Miyamoto, S. and Ihara, H. "Fuzzy control for automatic train operation system", IFAC Control in Transportation Systems, pp.33-39, 1983.

[YAS86]

Yasunobu, S. and Hasegawa, T. "Evaluation of an automatic container crane operation system based on predictive fuzzy control", Control - Theory and Advance Technology, Vol.2, No.3, pp.419-432, September 1986.

[ZAD73]

Zadeh, Lotfi A. "Outline of a new approach to the analysis of complex systems and decision processes", IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-3, No.1, pp.28-44, January 1973.

[ZAD85]

Zadeh, Lotfi A. "Syllogistic reasoning in fuzzy logic and its application to usuality and reasoning with dispositions", IEEE Transactions on System, Man, and Cybernetics, Vol.SMC-15, No.6, pp.754-763, December 1985.

[ZAD88]

Zadeh, Lotfi A. "Fuzzy logic", IEEE Computer, 0018-9162, pp.83-93, April 1988.

[ZAD92]

Zadeh, Lotfi A. "Calculus of fuzzy if-then rules and its applications", Applications of Artificial Intelligence X: Machine Vision and Robotics. SPIE, Vol.1708, pp.426-430, 1992.

[ZHA95]

Zhang, Yao; Sen, Pratyush and Hearn, Grant E. "An on-line trained adaptive neural controller", IEEE Transactions on Control Systems. Vol.15, No.5, pp.67-75, October 1995.

[ZIM87]

Zimmermann, H. J. Fuzzy Sets, Decision Making, and Expert Systems, Kluwer Academic Publishers, Lancaster, 1987.

[ZOL91]

Zolghadri, A.; Monsion, M. and Bergeon, B. "A supervised path planner", IEEE Proceedings, 7803-0078/91/0600-1710, pp.1710-1713, 1991.

[ZUE88]

Zuech, Nello. Application Machine Vision, John Wiley & Sons, New York, 1988.

[ZUE89]

Zeuch, Nello and Miller, Richard K. Machine Vision, Van Nostrand Reinhold, New York, 1989.

APPENDICES

Appendix A
TELECENTRIC LENS

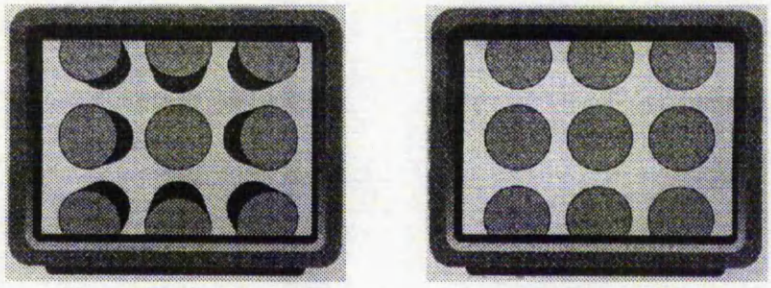
This appendix provides a description of the Telecentric Lens which can be employed to reduce viewing angle error and magnification error in a conventional vision system.

A.1 Introduction

The Computar 55 Telecentric is a 55mm f2.8 telecentric lens which reduces or eliminates *viewing angle error* and *magnification error* while providing high resolution and contract with low distortion. This compact, light weight lens is competitively priced and can be used with 1", 2/3", 1/2" and 1/3" format cameras. Options include 0.75X converter and 2X extender.

Machine vision has made the Telecentric lens popular because a conventional lens cannot accurately portray objects which are off axis to the lens, or at different distances from the lens. Viewing angle error and magnification error are inherent in conventional lenses, and these perspective distortions can create significant interpretation problems for software.

A true Telecentric lens, however, maintains a constant viewing angle at any point across the clear aperture of the objective lens. This characteristic of the Telecentric lens enables the machine vision system to generate images of objects which appear dimensionally accurate regardless of viewing angle or proximity to the lens. By eliminating perspective distortion, the Telecentric lens produces a dimensionally accurate image which is simple for software to interpret.



(a) coventional lens
(exhibiting viewing angle error)

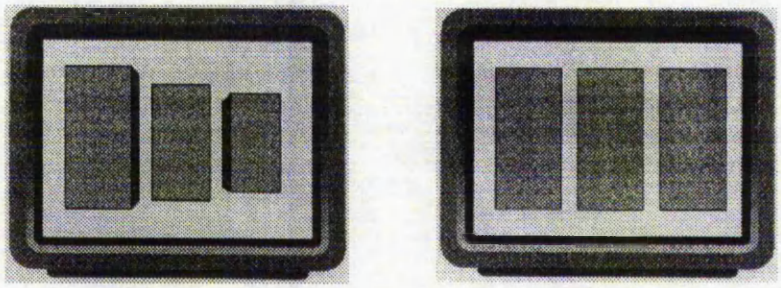
(b) Telecentric lens
(without viewing angle error)

Figure A-1: *Top view of nine identical cylinders using CCD camera*

A.2 Problem Description

Figure A-1 (a) shows an exaggerated top view of nine identical cylindrical objects as viewed by a conventional lens. Whether these objects are stationary or moving on a conveyor belt, the conventional lens will see each cylinder as dimensionally different, depending on its location relative to the axis of the lens. This viewing angle error is difficult for software to interpret.

Figure A-1 (b) illustrates the same top view of the same nine objects as seen by a Telecentric lens, which eliminates viewing angle error. Note that each cylinder appears to have the same shape and dimensions, regardless of viewing angle.



(a) coventional lens
(exhibiting magnification error)

(b) Telecentric lens
(without magnification error)

Figure A-2: *Front view of three identical boxes using CCD camera*

Figure A-2 (a) shows an exaggerated lateral view of three identical boxes which are positioned at different distances from a conventional lens. Note that the box located closest to the lens appears larger, while the box farthest from the lens appears to be smaller, while the two objects off axis to the lens also exhibit viewing angle error.

Figure A-2 (b) illustrates the same boxes as seen by a Telecentric lens. Note that all three boxes now appear to have the same shape and dimensions. That is the Telecentric Advantage.

A.3 Object Displacement vs. Image Size

An important problem with the conventional lens is that image scale magnification changes as a function of distance between object and lens. As the distance to a conventional lens increases, image size decreases. As this distance decrease, image size increases. Using a Telecentric lens, however, there is little or no apparent change in image size as an object dithers within given boundaries (see Figure A-3).

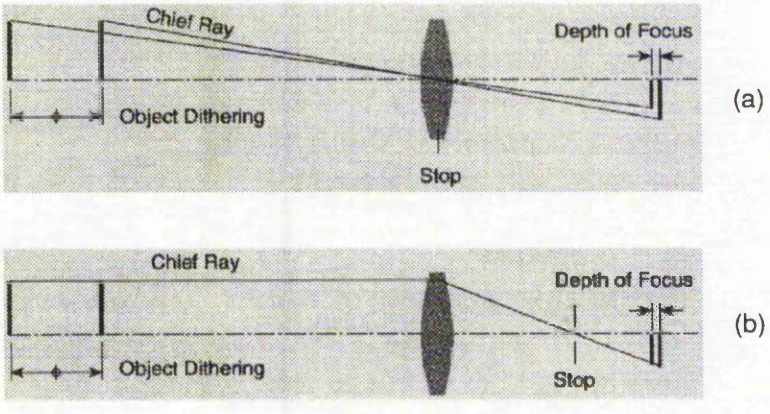


Figure A-3: *(a) Conventional lens with aperture stop inside lens;*
(b) Telecentric lens with entrance pupil at infinity and aperture stop behind lens

One of the most important benefits to be derived from the Telecentric lens is a reduction or elimination of magnification error or change in image size associated with variable distance between object and lens. But it is important to understand that three are inherent technical constraints to the application of this proven technology. First, this discussion assumes that dithering movement remains within the optical depth of field. Second, the lens will be truly telecentric or display no image movement only at close working distances and for objects which are smaller than the diameter of the first element in the lens. The Computar 55 Telecentric is truly telecentric when used at magnifications of 0.4X to 1:1. A field of view larger than the objective lens diameter will not yield true telecentricity, but will yield substantially less error than a conventional lens.

A.4 Field of View vs. Image Size

To understand Figure A-4, it is important to remember that system magnification is divided into optical and electronic components: Optical magnification is the ratio of the size of the image to the size of the object or part, while electronic magnification is a function of

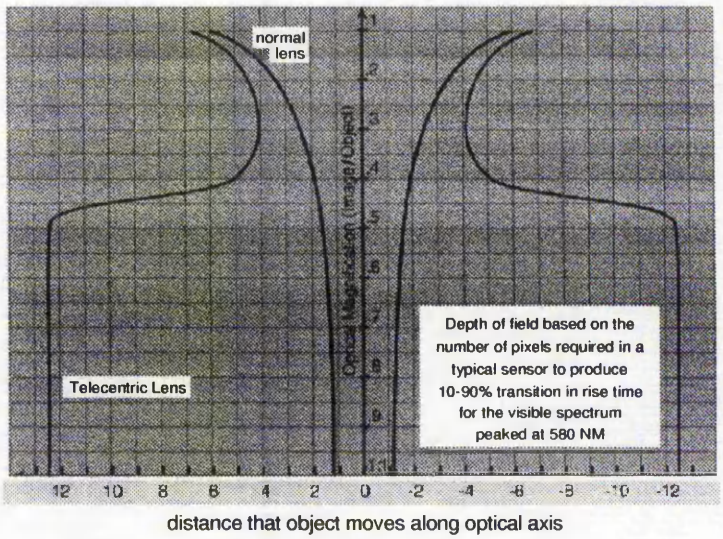


Figure A-4: *Object motion required to produce 1% error in image scale at various magnifications*

Working distance	*Lens*	*Camera format*
8"	55mm+0.75X	2/3"
11"	55mm	2/3"
11"	55mm+0.75X	1/2"
15"	55mm	1/2"
15"	55mm+0.75X	1/3"
19"	55mm	1/3"
22"	55mm+2.0X	2/3"
32"	55mm+2.0X	1/2"
37"	55mm+2.0X	1/3"

Table A-1: *Options available to reproduce a 2" wide object*

camera format and monitor size. If, when considering various camera formats, it is desirable to fill each sensor aperture equally, then the image size must be adjusted for each camera format.

With a 1/2" camera format, for example, 1 5" wide object can be reduced to the horizontal dimension of the sensor (6.4mm) by using: (a) the 55mm lens with object distance of 50", (b) this lens with 0.75X converter at 38" working distance, or (c) with the 2.0X extender at 95". Table A-1 displays the options available to reproduce a 2" wide object as large as possible on a monitor within reasonable physical constraints not dictated by camera format. Object size on the monitor will be 16" for a 20" monitor, or 9.5" for a 12" monitor.

It is helpful to note that the camera format can be changed without disturbing the balance of the inspection station. The information on a 2/3" sensor using the 55mm lens alone can be reproduced on a 1/2" sensor using the 55mm lens and a 0.75X converter. The same is true for 1/2" and 1/3" sensors.

Appendix B
PACER VMC HPGL

This appendix provides a description of the PACER VMC HPGL used to command the CNC machine in the project.

B.1 Introduction

PACER machines are controlled by an enhanced HPGL language via a DOS device driver or via a serial interface. This means that the PACER machine can be adequately controlled by even the most rudimentary CAD package, while allowing OEMs to configure more sophisticated packages to get the best from the system.

B.2 The Device Driver, PACER$

The PACER driver is called 'PACER$', and is a character device driver. There are three versions of the device driver, PACERVS1.EXE, PACERVS2.EXE and PACERDPR.EXE, that communicate with the controller via COM1, COM2 and shared memory respectively. All three drivers present an identical interface to DOS and Windows software. The driver is loaded by a line in CONFIG.SYS such as:

```
devicehigh=c:\pacer\pacerdpr.exe
```

It is envisaged that a CAD package could be installed so that the plotter is assumed to be on device 'PACER$', rather than the more conventional 'PRN' or 'COM1'. Thus, plotting a design from the CAD package causes it to be cut on the PACER machine without any further human intervention on the PC. For instance, plotting can be spooled from DOS

programs using the DOS PRINT command. The following commands may be added to AUTOEXEC.BAT:

```
lh print /D:PACER$
echo = > PACER$ (See section 'Driver Handshaking' for an explanation of this)
```

Likewise, plotting can be spooled from Windows by configuring your plotter as being on device PACER$, and allowing Print Manager to spool the job to the Pacer controller. Configuration, initialisation and depth control commands can be put in a file, and can be sent to the machine by a line in AUTOEXEC.BAT such as:

```
PRINT INIT.HPG
```

Other commands to change the behaviour of the machine can be copied to 'PACER$' between plots from the CAD system. The serial version of the Compact 800 will behave rather like an HPGL plotter, but with additional commands to perform toolsetting, depth and speed control, etc.

B.3 Installing the Card

The card should be installed in a free 16-bit slot in the PC. The 10-way ribbon cable supplied can be used to connect a standard 9-way serial port to the 10-way IDC connector on the card. If the cable is being connected to a serial port on the PC, the cable should be connected to the pinout marked 'PL8 IDE COM'. If it is being connected to some other, non-PC serial port, the connector can be plugged into 'PL4 TERMINAL' to behave as a DTE serial port, or 'PL8' to behave as a DCE serial port. In both cases, the red strip should be towards the metal plate end of the card.

Note that if the card has the DPR chips installed, and the PACERDPR driver is being used, then there is no need to connect the serial cable. However, any DOS memory manager, (such as QEMM, EMM386, 386MAX) need to be instructed to exclude the memory region EF00-EFFF, as this is where the DPR memory resides. This is done by appending 'X=EF00-EFFF' to the memory manager line in CONFIG.SYS.

B.4 Link Settings

JP1 (Reset). The 68000 controller will reset when these pins are shorted together. A reset switch can be fitted, but is only useful for R&D.

JP2 (ROM type). With pins 2-3 shorted, this selects 27256 ROMs. With pins 1-2 shorted, this selects 27512 ROMs. The VMC software is currently supplied on 27256s, and this link MUST NOT be moved from pins 2-3.

JP3/JP4 (RAM type). There are 4 possible settings for these two links. The meaning is as follows:

JP3:1-2, JP4:1-2 DPS5128P (1Mb)

JP3:1-2, JP4:2-3 TC55100PL (256K)

JP3:2-3, JP4:1-2 DPS2568P (512K)

JP3:2-3, JP4:2-3 43256-85 (128K)

JP5 (Watchdog enable). This link MUST be left open, as there is currently no software support for the watchdog function.

JP 6/7 (Machine type). These links are labelled 'A' and 'B'. Each link can be in one of two positions, labelled '0' and '1'. The meanings are as follows:

A=0, B=0 Compact 800

A=1, B=0 K range defaults

A=0, B=1 HD range defaults

A=1, B=1 Not defined, but may be 'test mode' in future

B.5 Testing the Card

A simple test that the card is working can be achieved by running the DOS program 'VMC_CTRL.EXE', supplied with the card. With no arguments, this program enquires the card status every second, and displays it on the screen. PACER Systems recommend that the DOS VMC program is installed, and used to test the system. To install VMC, the minimum requirements are:

1) The driver, PACER$, installed as above,
2) A directory C:\VMC, containing: VMCX.EXE, SETUP.FMT, LANGUAGE.FMT, PACER.PCR, PACER.HPG
3) A directory, C:\VMC\PACER, where user jobs can be stored.

HPGL has essentially two groups of commands. One group consists of two-letter mnemonics and arguments. Most plotting is done with this set. These commands are queued in a large buffer. The second group start with an ESCape character, and are executed immediately. These 'queue-jump' the other commands, and are used for interactive/real-time commands.

As a job progresses, the machine status will change. Each time that happens, the current status will be transmitted back. As this type of message is asynchronous and 'unexpected', the message will commence with a BELL character (Ctrl-G), and terminate with a newline (Ctrl-J). Some commands will cause text to be returned. In this case the text is expected, and the text will NOT start with a BELL. It will, however, still be terminated with a newline character.

B.6 Commands

B.6.1 Immediate commands

HPGL specifies a dot between the 'ESC' and the command letter. In the Pacer implementation, this dot will be optional.

Command	HPGL meaning	Pacer meaning
ESC.A:	Output m/c type & ver	Output ROM version
ESC.O:	Output status	Output status (Pacer format)
ESC.R:	Reset plotter	Reset controller
ESC.K:	Flush input buffer	Flush input buffer

Immediate commands (Pacer extension)

ESC.C 0:	Pause
ESC.C 1:	Continue
ESC.C -1:	Toggle Pause/continue state
ESC.D ddd,ttt:	Set Immediate depth of cut. ddd in mm. (Cleared by IN;) Set thickness to ttt in mm. (Not cleared by IN;)
ESC.F 0:	Seek datums (Only available if machine has datum switches - See ESC.#)
ESC.F 1:	Auto Toolset (Only available if machine has an auto toolset plunger - See ESC.!)
ESC.F 2:	Initiate manual toolset
ESC.F 3:	Confirm manual toolset (Set Z bed position)
ESC.F 4:	Set XY origin
ESC.F 5:	Goto XYZ origin
ESC.F 6:	Empty swarf (if possible) and park (Only available if parking has been enabled - See ESC.@)
ESC.G 0, nnn:	Move Z axis Up by nnn millimetres (Note 1)

ESC.G 1, nnn: Move Z axis Down by nnn millimetres (Note 1)

ESC.G 2: Move Z axis to bed position

ESC.G 3: Move Z axis to 'Pen Up' position

ESC.G 4: Move Z axis to 'Park' position

ESC.G 5, nnn: Move X axis Left by nnn millimetres (Note 2)

ESC.G 6, nnn: Move X axis Right by nnn millimetres (Note 2)

ESC.G 7, nnn: Move Y axis Forward by nnn millimetres (Note 2)

ESC.G 8, nnn: Move Y axis Backwards by nnn millimetres (Note 2)

ESC.G 9: Immediate splindle motor on (engineer's use only).

ESC.G 10: Immediate splindle motor off (engineer's use only).

ESC.G 11: Move Z axis to 'Pen Down' position.

ESC.G 12, nnn: Move X axis Right and Y Forward by nnn millimetres (Note 2)

ESC.G 13, nnn: Move X axis Left and Y Forward by nnn millimetres (Note 2)

ESC.G 14: Immediate swarf vacuum on.

ESC.G 15: Immediate swarf vacuum off.

ESC.G 16: Immediate Laser Enable on.

ESC.G 17: Immediate Laser Enable off.

ESC.V f,p: Set velocities (now) where:

'f' is feed rate, 'p' is plunge rate all in metres per minute. Note that any parameters omitted will be left unchanged.

ESC.P c,f,p: Set multipass options where:

'c' is the maximum cut in mm, 'f' is the finishing cut in mm, and 'p' is the number of finishing passes. (See 'MP' below.

ESC.L nnn: Set laser strike delay to nnn tenths of a second.

= This enables the driver into handshaking. (See section below)

Notes: (1) Z axis movements default to 6 mm if 'nnn' is omitted.

(2) X and Y axis movements default to 100 mm if 'nnn' is omitted.

B.6.2 Machine configuration commands

The following commands are required to be sent to the controller whenever the controller starts from cold (Reset command, power on, hardware reset). This is because the controller does not have any non-volatile memory, and needs to be told about the machine that it is controlling.

ESC.# xyc, zc, c,m: Set conversion factors. 'xyc' is the number of horizontal machine steps per millimetre. 'zc' is the number of vertical machine steps per millimetre. 'c' is 1 if the machine has a Compact800-style joystick control panel. (zero otherwise). 'm' is the maximum intstep used in interpolation. The default value is 60, and should ONLY be changed on specific instructions from Axiomatic Technology Ltd.

ESC.@ xp,yp,xs,ys: Set PARK position to (xp,yp), and swarf bin position to (xs,ys), and enable park command. All in mm. 'xs' and 'ys' ommitted if machine does not have a swarf bin.

ESC.$ xh,yh,x1h,y1h: Set auto-homing microswitch positions in mm. (xh,yh) is the position of the microswitches with respect to the router spindle. (x1h,y1h) is the position of the microswitches w.r.t. the auxilary pen/knife holder. (x1h and y1h not yet implemented).

ESC .! xt, yt,zt: Set Auto toolset plunger position to (xt,yt,zt), in mm, and enable auto toolsetting.

ESC.X xys,zs,a,zt,dr: Set velocities (now) where:

'xys' is horizontal slew rate, 'zs' is vertical slew rate, 'zt' is the toolset speed, dr is the driveoff speed for seeking datums, all in metres per minute. 'a' is acceleration in percent of G. Note that any parameters omitted will be left unchanged.

ESC.W nnn: Set park position to nnn millimetres above the bed (default 50mm).

ESC.& x1,y1,x2,y2: Set hard clip limits (Bed area).

B.6.3 Enquiring from the machine

ESC.*:	Perform ROM checksum. Return '* 0 ^J' for success, '* n ^J' for failure, where 'n' is a non-zero integer.
ESC.D:	Enquire thickness and depth of cut. Returns:
D d t ^J	'd' is the immediate depth of cut, 't' is the thickness, both in mm.
ESC.V:	Enquire velocities. Returns:
V f p ^J	'f' is feed rate, 'p' is plunge rate both in metres per minute.
ESC.X:	Enquire velocities. Returns:
X xys zs a ^J	'xys' is horizontal slew rate, 'zs' is vertical slew rate, all in metres per minute. 'a' is acceleration in percent of G.
ESC.P:	Enquire multi-pass parameters. Returns:
P c f p ^J	'c' is the maximum cut, 'f' is the finishing cut, both in mm, and 'p' is the number of finishing passes.
ESC.T:	Enquire Tooltype and diameter. Returns:
T 't' diam ^J	't' is the tool type, one of 'P' (pen), 'K' (knife), 'R' (router), 'E' (engraving head), 'M' (Miller), 'L' (Laser). Diam is tool diameter in mm.
ESC.O:	Enquire status. Returns:
O d s i c w p q r^J	'd' is the drive status, 's' is the system status (See 'Status Returned from Machine' for format). 'i' is the state of the hardware input bits as a hexadecimal number. The bits are as follows:

0001 Safety Circuit switch

0002 Offline button

0010 Z Datum switch (Toolset button on Compact 800 range)

0020 Toolset switch

0040 X Datum switch

0080 Y Datum switch

Drive state is: 0: Stopped, 1: Seeking Datums, 2: Auto Toolset, 3: Moving (Cutting or slewing).

'c' is the time since the controller was started in tenths of a second. 'w' is the text 'cold' until the 'ESC #' command is used, and 'warm' subsequently. 'p' is the number of outstanding paths to cut. 'q' is the number of paths since the last IN command. 'r' is the ram size installed in the card.

ESC.Z: Enquire XYZ position on millimetres relative to origin. Returns:

Z X=1234.56, Y=1234.56, Z=123.45^J

B.6.4 Vector commands

PU;	Lift pen.
PU x,y;	Lift pen and move to (x,y).
PD;	Pen down.
PD x,y;	Pen down and draw to (x,y).
PD x,y,x,y,...,x,y;	Pen down and draw polyline.
PA;	Set absolute mode.
PA x,y;	Set absolute mode and move/draw to (x,y).
PA x,y,x,y,...,x,y;	Set absolute mode and move/draw polyline.
PR;	Set Relative mode.
PR x,y;	Set Relative mode and move/draw to (x,y).
PR x,y,x,y,...,x,y;	Set Relative mode and move/draw polyline.
AA x,y,ang;	Draw arc absolute centre (x,y), angle ang.
AA x,y,ang,ct;	Draw arc absolute centre (x,y), angle ang, ct ignored.
AR x,y,ang;	Draw arc relative centre (x,y), angle ang.
AR x,y,ang,ct;	Draw arc relative centre (x,y), angle ang, ct ignored.

AT x1,y1,x2,y2,x3,y3;Three-point arc. (HPGL/2).

RT x1,y1,x2,y2,x3,y3;Relative Three-point arc. (HPGL/2).

CI rad;	Draw circle radius rad.
CI rad,ct;	Draw circle radius rad, ct ignored.

B.6.5 Configuration and control commands

SP; or SP0;	End job. This is required.
SP pen;	Select specified pen.
IN;	Lift pen, select solid line type, absolute mode, clear immediate depth of cut.
DF;	Lift pen, select solid line type, absolute mode.
SO;	Set origin to the current point. (Compatible with the Aristo plotter SO command.)
IWx1,y1,x2,y2;	Input Window (clipping).
SC;	xmin,factor,ymin,factor,2; Scale plot by factor
NR;	Not Ready - go offline until online pressed.
OA;	Output actual pen position in machine units.
OC;	Output commanded pen position in HPGL units.
OF;	Output factors for X & Y (plotter units per millimetre).
OI;	Output identification.
OH;	Output hard clip limits, ie bed area. (in millimetres)
OS;	Output current scale command
OW;	Output Window (in millimetres)
TT type,diam;	Set tooltype for all pens. Type can be: 'P' (pen), 'K' (knife), 'R' (router), 'E' (engraving head), 'M' (Miller), 'L' (Laser). Diam is tool diameter in mm.
STn;	Perform Self-test number 'n'. Currently self-tests 1-4 are defined. Test 1 is the text 'Pacer Systems Limited'. Test 2 is the word 'PACER'. Test 3 sets pen depths for a variety of depths suitable for cutting from 12mm thick material, and does a simple, multidepth job. Test 4 just set the pen and

depth parametes as above, but does not actually cut anything. 'IN' and 'SC' commands can precede these commands.

B.6.6 Configuration commands (Pacer extension)

DE pen,d1,d2,type,dia; Set depth range for 'pen', and define its type. Type can be: 'P' (pen), 'K' (knife), 'R' (router), 'E' (engraving head), 'M' (Miller), 'L' (Laser). Dia is tool diameter in mm. This will override the last TT command.

DE pen,d1,d2; Set depth range for 'pen'

DE pen, depth; Set depth for 'pen'

DE 0, depth; Set depth for all pens. All depth values are in mm.

DE pen; This will return pen information for pen 'pen'.

DE; This will return all the current pen information.

TH thickness; Set material thickness in mm.

HO pen, min, max; Hole control for 'pen'. Circles with radius less than 'min' get discarded, while circles with radius between 'max' and 'min' get drilled as holes.

(see note on depth control for more details of the DE commands)

FL; Flush buffer. All commands before this are discarded (unless already executed).

MPx; Multipass suspend/resume. x=1 for resume, x=0 for suspend.

SR r10,r15,r20,r30,r40,r50,r60,r70,r80;

Set Radii for speed control.

The machine needs to slow down round arcs, especially small arcs, or it will stall. This command determines how much the machine slows down for each possible radius. Arcs with radius below 'r10' (in mm) will be cut at not more than 10% of the current slew speed. Arcs with radius between 'r10' and 'r15' will be cut at not

more than 15% of slew speed. Likewise, Radii 'r15' to 'r20' at 20%, 'r20' to 'r30' at 30%, 'r30' to 'r40' at 40% and so on. Arcs with radius more than r80 will be cut at full speed. Note that these values **must** be in order of increasing radius. Any value unset will take the previous value multiplied by 1.5. The default values are: 2, 3, 5, 8, 10, 15, 25, 45, 70mm.

SA a10, a30; Set angle for speed reduction.

The machine needs to slow down at corners between vectors. If the angle is sufficiently small, less than 'a30' (in degrees) then the machine can continue cutting at full speed. However, if the angle is greater than 'a30' but less than 'a10' (in degrees) then the cutter will not excede 30% of slew speed. Also, if the angle is greater than 'a10' but less than 90 then the cutter will not excede 10% of slew speed. If the angle at the intersection is greater than 90 degrees, then the cutter will instantaneously stop at the corner. 'a10' must be greater than 'a30'. The default values are 20 degrees for 'a10' and 10 degrees for 'a30'.

DB level; Sets the debug level. This controls the number of messages that the controller produces. The level is treated as an 8-bit word of binary flags. Thus, a level of 6 indicates that flags 2 and 4 are set. the meanings are as follows:

0: No debug messages. Errors, warnings and status only.

1: Print design vectors after they have been parsed.

2: Messages from Machine vector generation.

4: Print vectors at cutting/interpolation stage.

8: Display ramp table when acceleration changed.

16: Print movement requests to Drive controller

When the controller is run from the Pacer machine control software, these messages will end up in the VMC.ER log file. This information is extremely useful to Pacer staff when trying to diagnose a problem. Note that debug flags 1 and 4 will have an adverse affect on machine performance due to the very large amount of data being transmitted.

B.6.7 Commands with different meaning

Command	HPGL meaning	PACER meaning
FS force,pen;	Set force for 'pen'	See note on Depth Control
FS force;	Set force for all pens	See note on Depth Control
VS speed, pen;	Set speed for 'pen'	See note on Feed Rate
VS speed;	Set speed for all pens	See note on Feed Rate

B.7 Depth Control

Note that there is no standard HPGL command for depth. We could invent one, but we would then have no support for it in the CAD systems that are out in the field. The solution is based on the FS (Force Select), SP (Select Pen) commands, and a single new command DE (Depth control). The DE command could be used in a header file at the start of a job, or may be built up in an interactive OEM program.

HPGL supports 8 pens, with a force value in the range 1-8 for each pen. The default force for each pen is 1 (minimum force). In its simplest form, DE sets an absolute depth for each pen number. This would allow RoboCAD users to plot each of 8 layers in a different pen, and thus get the 8 layers cut at 8 different depths.

For example: DE 1, 3.2; DE 2, 10.55;

This would mean that pen 1 cut at a depth of 3.2mm, and pen 2 at 10.55mm. A more sophisticated form of the DE command would set for each pen a depth for force=1 and a depth for force=8. Other forces would be interpolated.

For example, DE 1, 3.3, 5.05;

This would mean that at minimum force, pen 1 cut at 3.3mm, while at maximum force pen 1 cut at 5.05mm. The the commands 'SP1; FS 1,1;' would set a depth of cut of 3mm for the next path, while 'SP1; FS 1, 4;' would set the force for pen 1 to 4. Interpolating between 3.3mm and 5.05mm, 4 is 3/7 of the way up the scale, and would thus set a depth of (3.3 + (3/7)*(5.05-3.3)) mm. The actual depth cut is the sum of the depth associated with the current pen, and the current immediate depth, set with the 'ESC D' command.

B.8 Feed Rate Control

HPGL associates a feed rate (velocity) with each pen, and this will be our aim in the long term. However, our current controller has no facility for associating a feed rate with a path - feed rate commands are actioned immediately.

In the interim, Pacer recommend the use of the ESC.V command to set the feed rate. When implemented, the VS command will be actioned soon after it is received by the device driver, rather than being queued. Also note that the interim implementation will ignore the pen number on the VS command. If the buffer is empty and the machine is idle, then VS and ESC.V are identical.

B.9 Status Returned from the Machine

The version string can be obtained by ESC.A or OI. The string returned is as folllows:

$Revision xx.xx $ description, copyright ^J

where 'description' describes the board, and 'copyright' is the PACER copyright message. For example:

$Revision 6.1 $ Pacer-VMC HPGL controller, (c) Axiomatic Technology 1994 ^J

There are two types of status that are returned asynchronously. Firstly, there is the system status. This is a 16 bit word of binary flags. From the flags, it is possible to deduce what commands are available. The flags are transmitted back as a single hexadecimal unsigned integer in the range 0000 to FFFF. The values associated with each flag are as follows:

0001: Safety circuit broken.

0002: XY position known (Position has not been lost since last seek datums).

0004: Tool height known (Position has not been lost, or tool changed, since last toolset).

0008: Data buffer full (Set when less than 2K available; Cleared when more than 4K available).

0010: Active: machine is moving.

0020: Offline: Machine has acknowledged offline request button.

0040: Paused: Machine has received a pause command.

0080: Parking: Machine is moving to the park position.

0100: Seeking: Machine is seeking datums.

0200: Autotoolsetting.

1000: Machine has Compact-800 style joystick panel.

2000: Autotoolset has been enabled.

4000: Park has been enabled.

8000: Machine has Datum switches.

The format of the message returned is as follows:

^G S hhhh ^J

Secondly, there is en error message. The format of the message returned is as follows:

^G E n xxxx ^J

where 'n' is the error number. Possible numbers are:

1: Bad HPGL two-letter mnemonic. 'xx' contain the offending two letters.

2: Serial transmission error. xx is a hexadecimal number containing the error bits from the serial chip.

3: Movement discarded: machine busy.

4: Command discarded: Feature not enabled.

5: Bad ESC command. 'x' contains the offending command character.

6: Command discarded: Safety circuit broken.

7: Movement discarded: machine offline.

8: Interrupt error. This is fatal, and causes the controller to restart.

9: Three-point arc error.

10: Exception 'n' at 'address'. Report all instances of this error to PACER Systems.

11: Insufficient buffer space for self-test.

12: Cannot park Z, clear-check, up-check, down-check while not toolset

13: Path too big to fit in memory

14: Material Thickness too great for current tool position

B.10 Driver Handshaking

Normally, it is the responsibilty of the application to handshake with the controller by using the '0008' bit in the status word returned from the controller. This allows immediate

commands to be sent, even though the buffer is (nearly) full. However, if the driver is to be driver by the DOS 'PRINT' command, or directly from a CAD package that does not have the facility to send other immediate commands, (such as Pause/Continue), then the driver must handshake with the controller to prevent the controller data buffer overflowing. This is done by sending a single '=' character to the driver. Note that this only applies to the serial version as the DPR version cannot overrun data.

B.11 Restrictions

The data is assumed to have been through PPD. That is, the data must be linked, sorted and nested ready for cutting. Neither the device driver nor the controller will change the order in which the vectors are processed. Initially, the VS command will be interpreted immediately, as the the current generation of PACER machines have no concept of a 'queued' change in feed rate.

Note that 'AT' and 'RT' are HPGL/2 extensions to the HPGL language, and that he job needs to be terminated by an 'SP0' or 'SP' command.

B.12 Controller Interface

The interface to the machine is via four 'C' functions, as follows:

```
void controller_immediate_write (char * command);
```

'command' is a null-terminated ASCII string that starts with an escape character. This function may take a few seconds before it returns, if the immediate queue is full. Only one immediate command should be in each call to this function.-

```
int controller_vector_write (char * command, int wait_flag);
```

'command' is a null-terminated ASCII string containing HPGL two-letter commands, seperated by a semicolon. The maximum length for 'command' is 1K. If 'wait_flag' is TRUE, the routine will wait until the data has been transmitted (suspending the task in the meantime). It will return ther number of bytes actually transmitted.

```
int controller_response_read (char *buffer, unsigned buff_len, int wait_flag);
```

'buffer' is the address of a memory block into which the response will be put. The available length of 'buffer' should be stored in 'buff_len'. If 'wait_flag' is TRUE, the routine will wait until a response arrives (suspending the task in the meantime), and will eventually return TRUE. If 'wait_flag' is FALSE, the routine will return immediately, returning TRUE if a message was available and has been put into 'buffer'.

```
int controller_async_read (char *buffer, unsigned buff_len, int wait_flag);
```

'buffer' is the address of a memory block into which any asynchronous messages from the controller will be put. The available length of 'buffer' should be stored in 'buff_len'. If 'wait_flag' is TRUE, the routine will wait until a response arrives (suspending the task in the meantime), and will eventually return TRUE. If 'wait_flag' is FALSE, the routine will return immediately, returning TRUE if a message was available and has been put into 'buffer'.

Appendix C
SAMPLES OF SMP FOLLOWING USING THE 2VMETHOD AND THE 3VMETHOD

This appendix provides various samples taken from the experiments of SMP (*Spring Mounted Pen*) following applying the 2VMethod (together with the Piecewise Error Compensation Algorithm) and the 3VMethod.

C.1 Two-Vector Method (2VMethod)

Figure 2V-A-1 to Figure 2V-D-4 illustrate the results of SMP following using the 2VMethod together with the Piecewise Error Compensation Algorithm. Figure 2V-A/D-1 (i.e. Figure 2V-A-1, Figure 2V-B-1, Figure 2V-C-1, and Figure 2V-D-1) are the patterns which do not add any correcting commands to the scanned paths. Figure 2V-A/D-2 to Figure 2V-A/D-3 are the patterns produced through the learning processes. Figure 2V-A/D-4 are the final results of the path-following-processes using 2VMethod. Almost all the path-following-errors caused by the spring are successfully eliminated.

In contrast to the SMP following utilised 2VMethod, some sample patterns using 3VMethod are illustrated and described below.

C.2 Three-Vector Method (3VMethod)

Figure 3V-1 to Figure 3V-6 shows the final results of using the 3VMethod for the SMP following process. By applied the 3VMethod about 60 to 80 percents of path-following-errors can be removed. Compared with the SMP following applied 2VMethod, it is obvious to see that the 2VMethod can produce excellent results better than the 3VMethod.

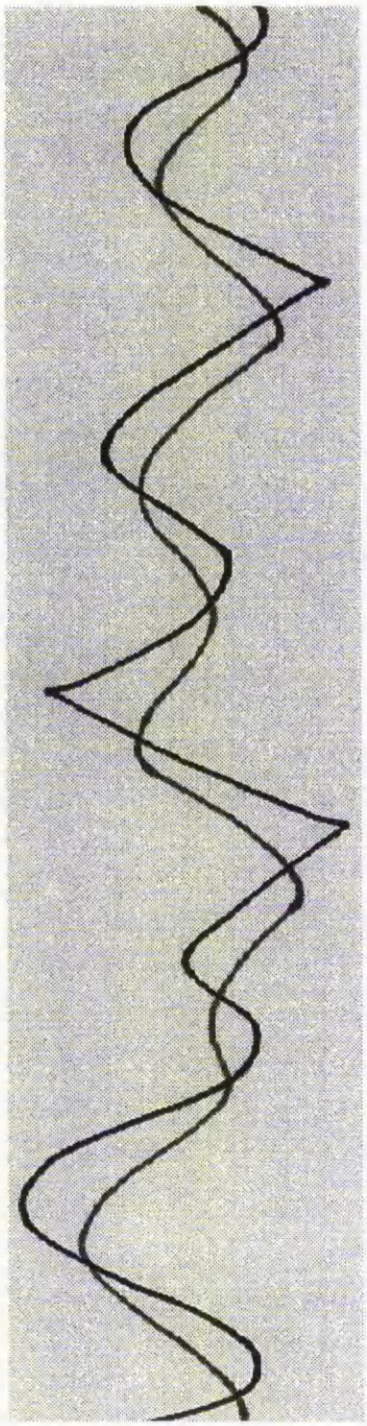


Figure 2V-A-1: Frame Zero

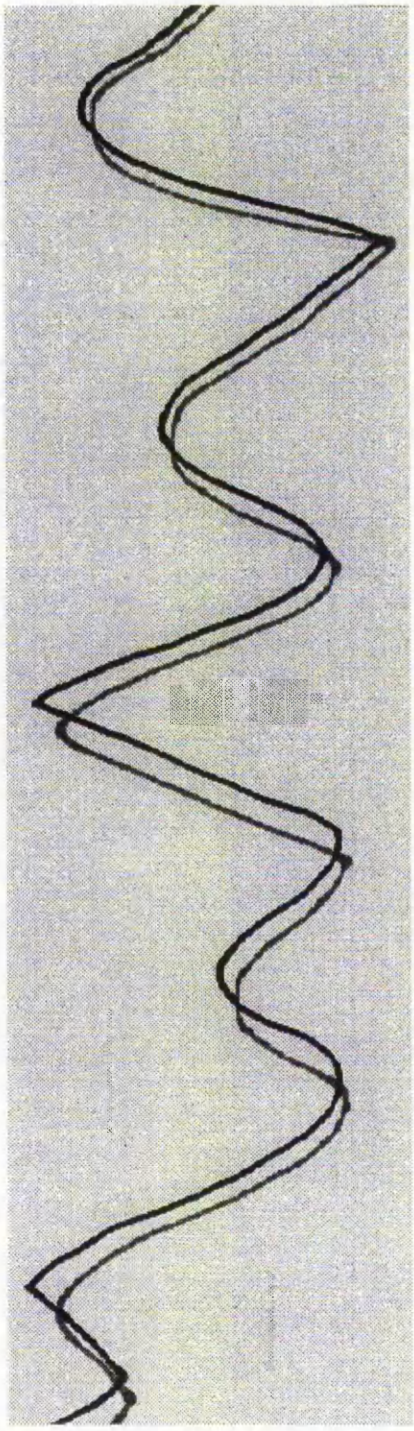
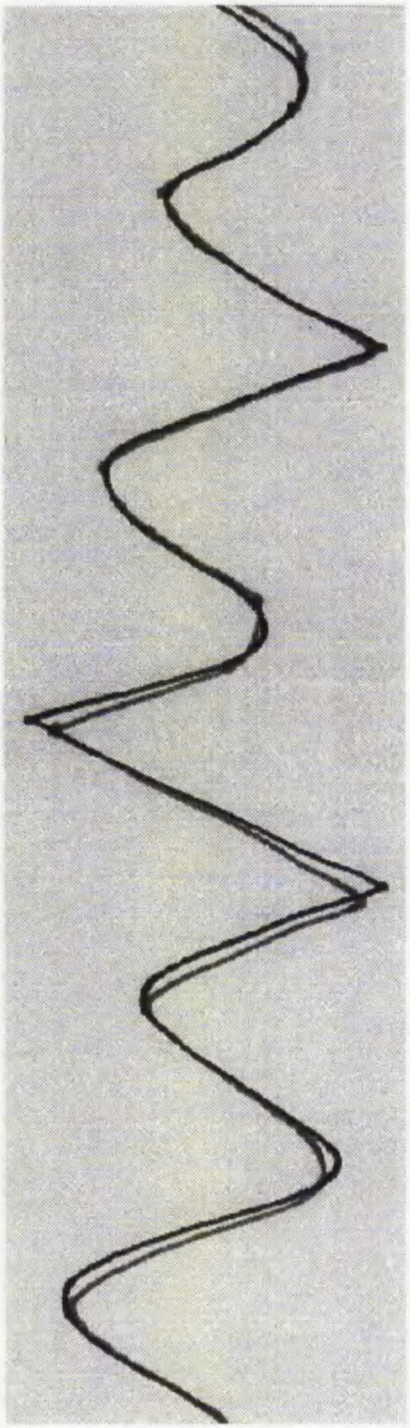
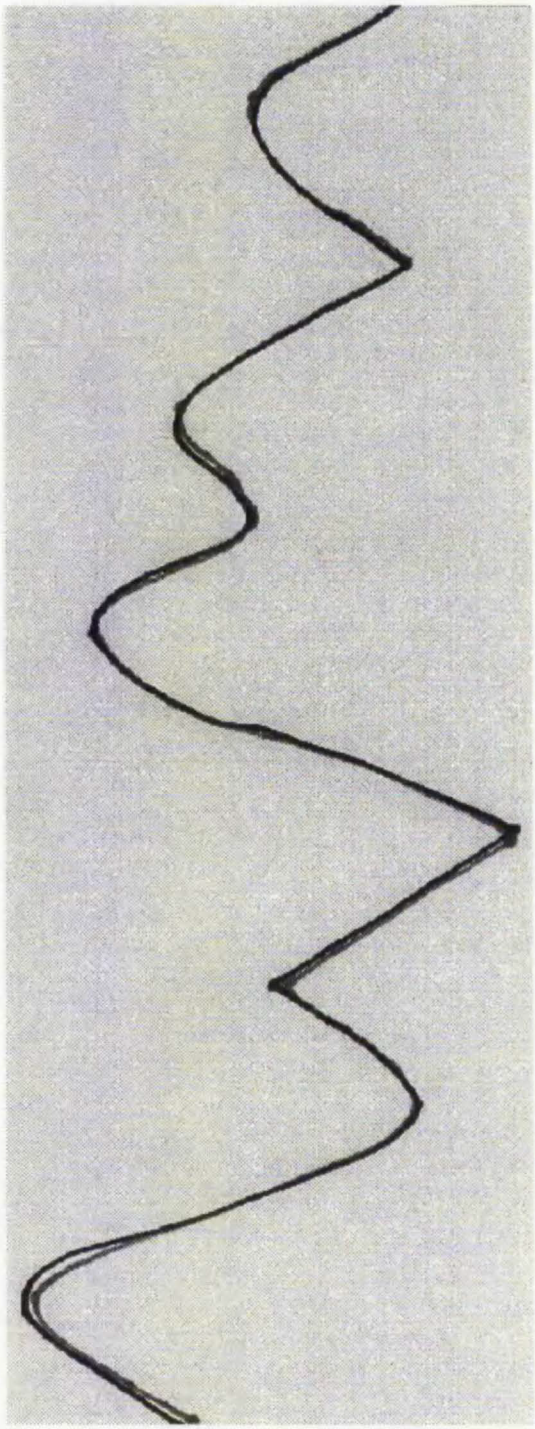
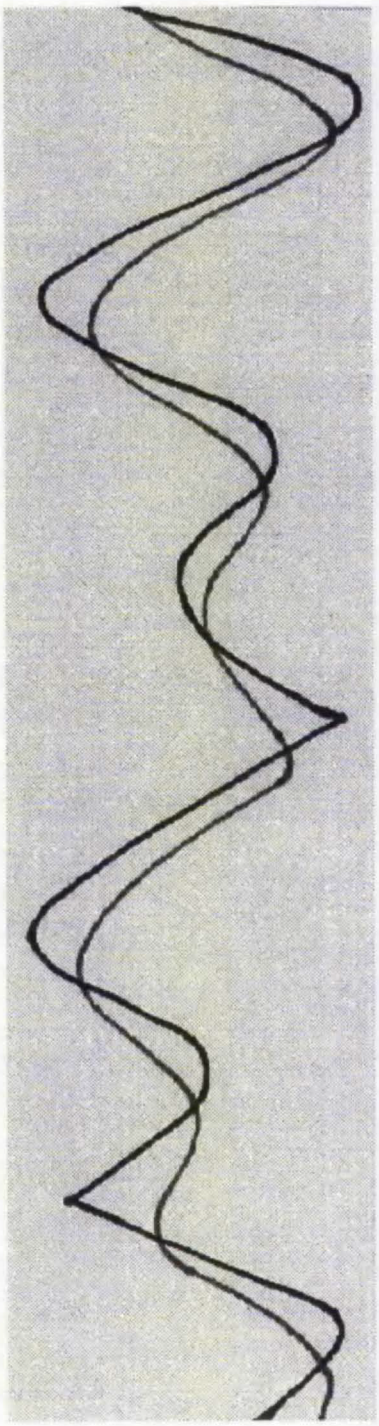
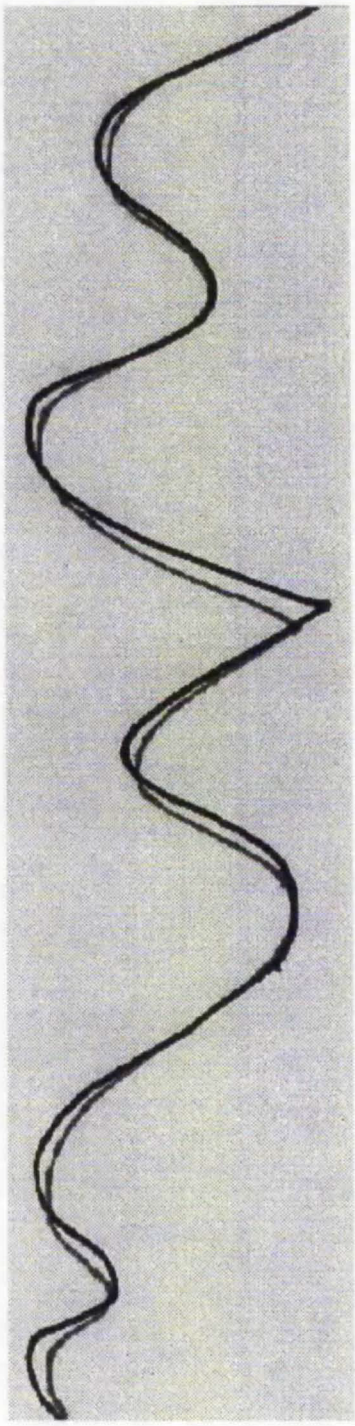


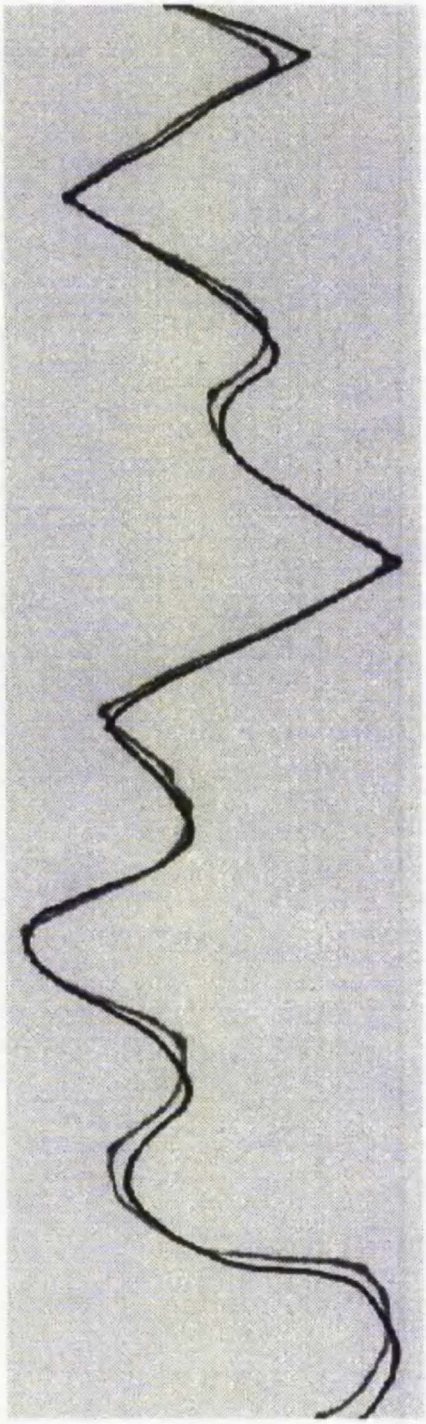
Figure 2V-A-2: Frame One

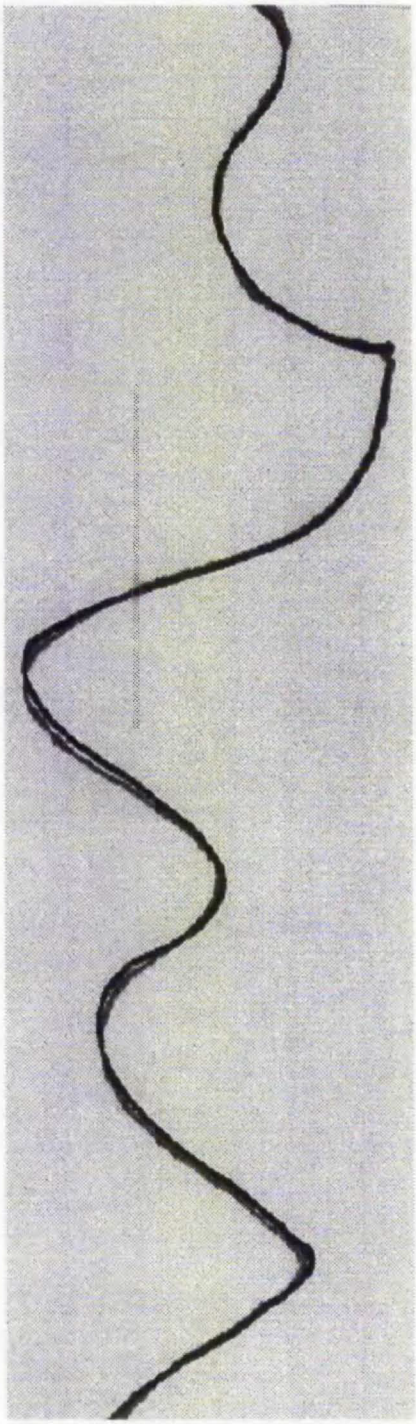












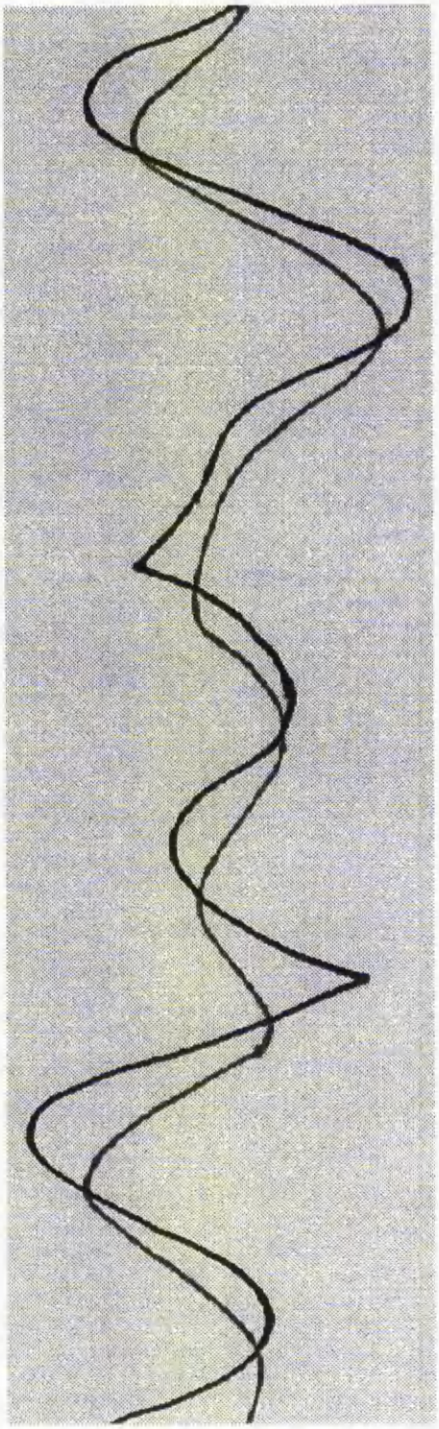


Figure 2V-C-1: Frame Zero

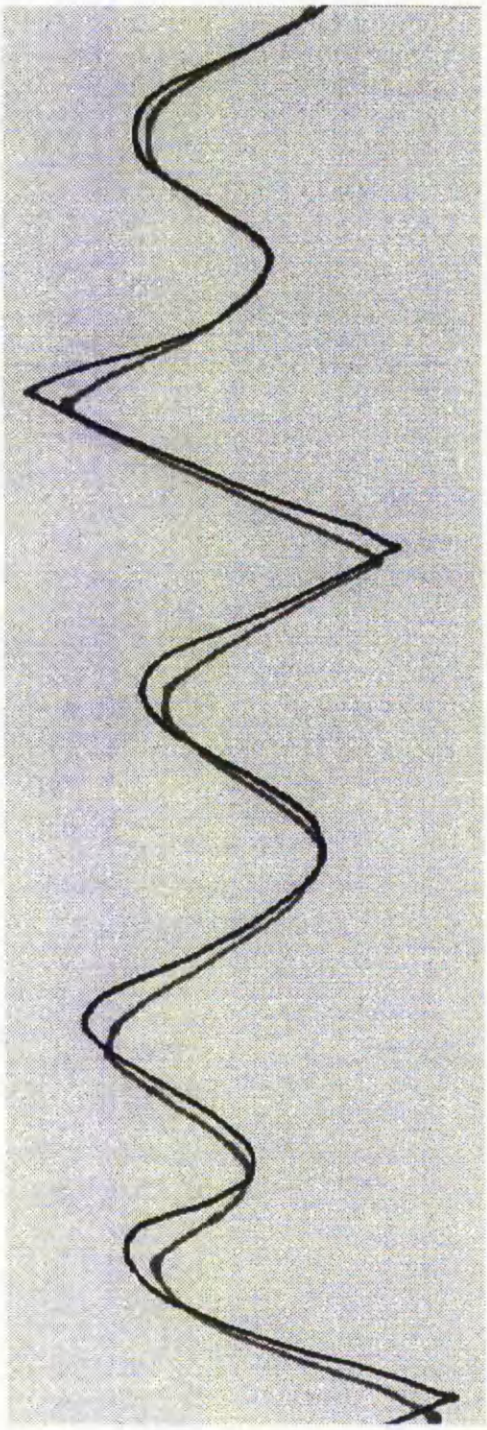


Figure 2V-C-2: Frame One

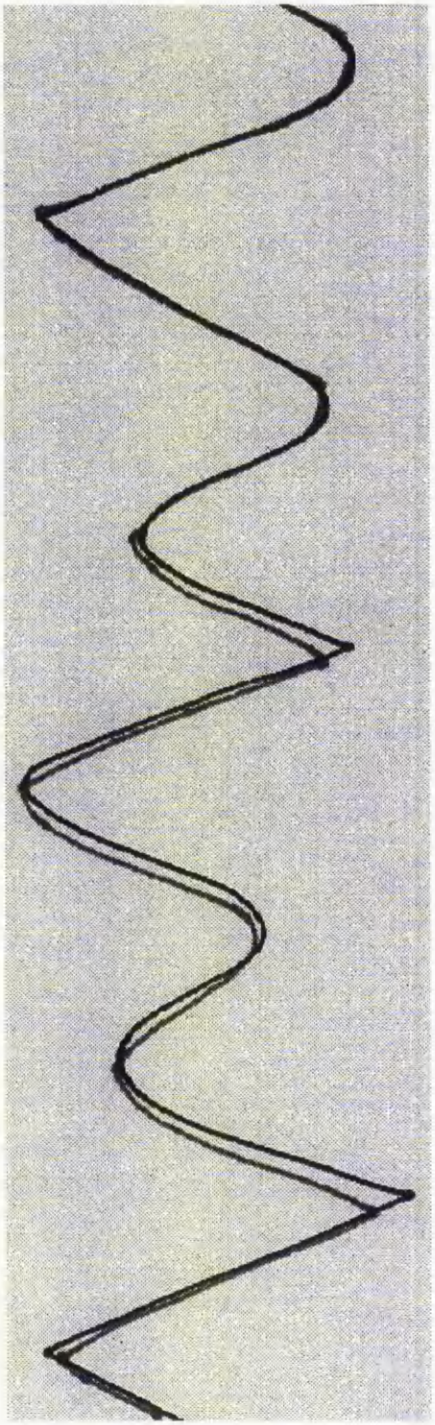


Figure 2V-C-3: Frame Two

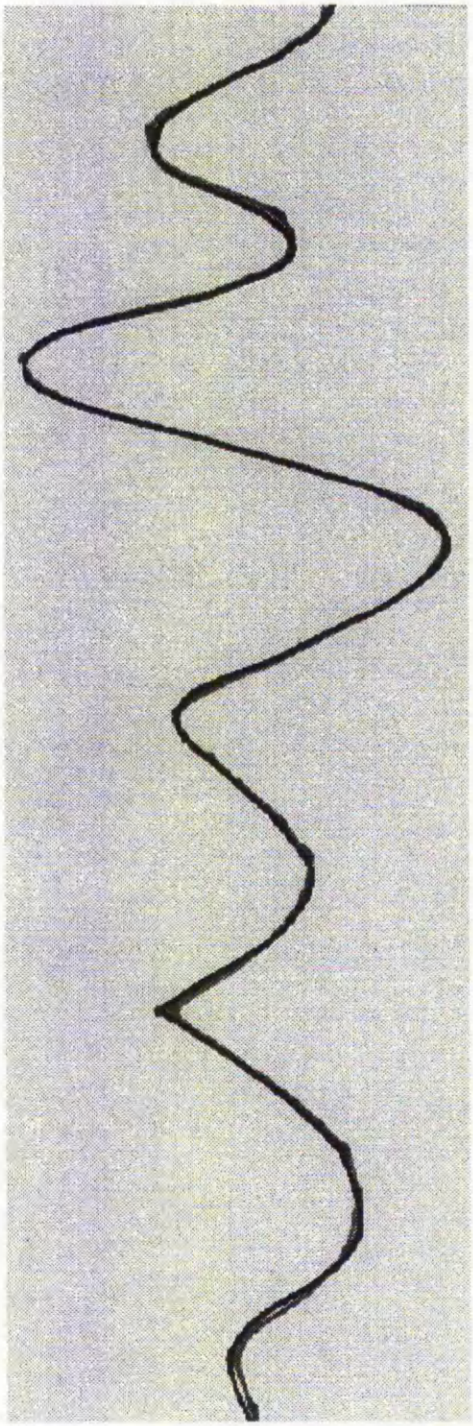


Figure 2V-C-4: Frame Three

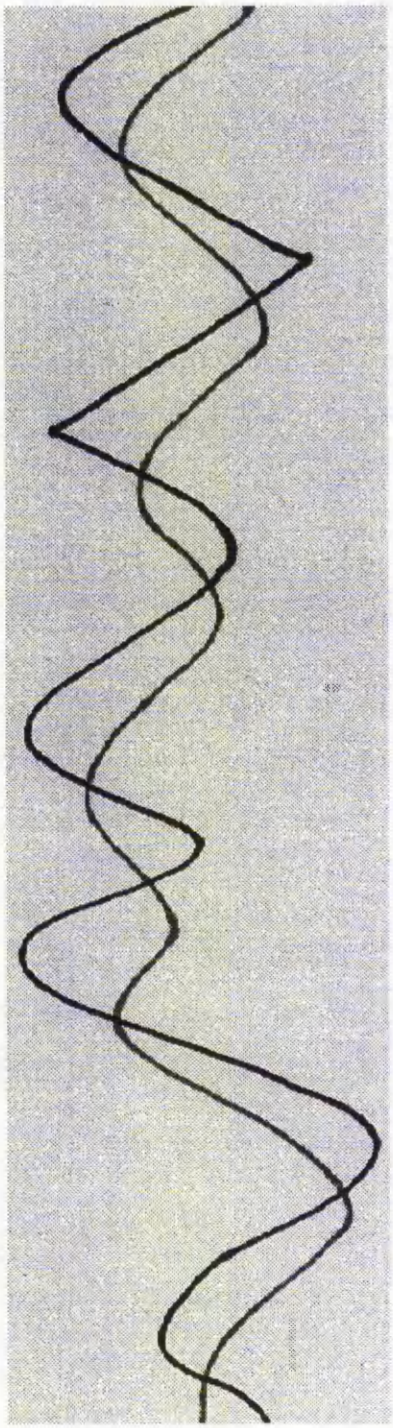


Figure 2V-D-1: Frame Zero

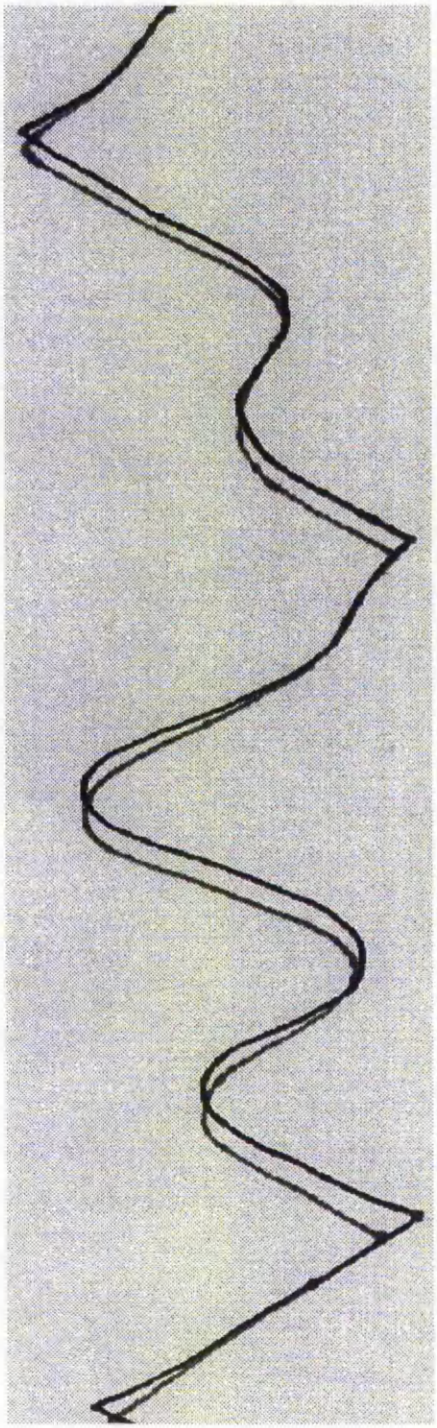


Figure 2V-D-2: Frame One

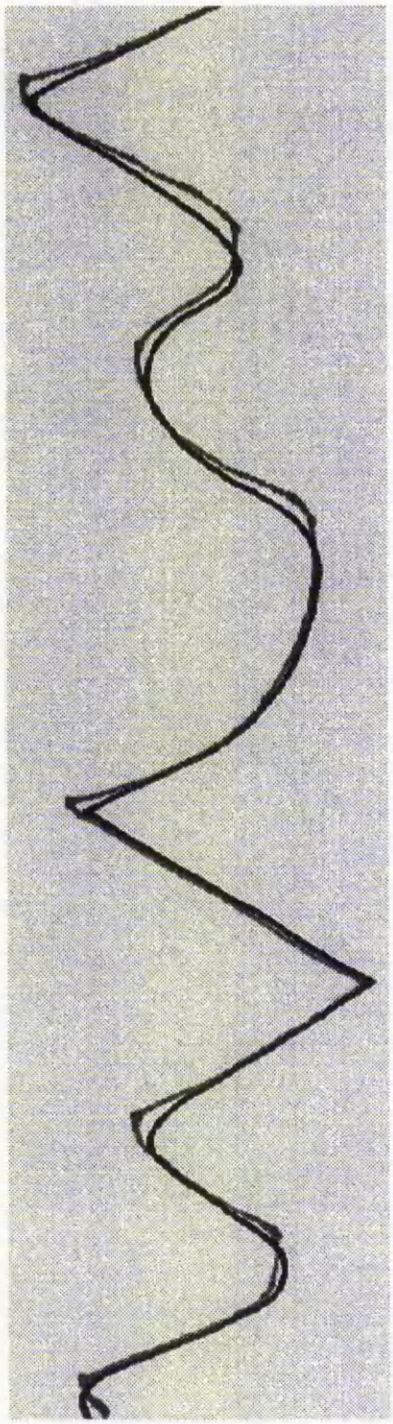


Figure 2V-D-3: Frame Two

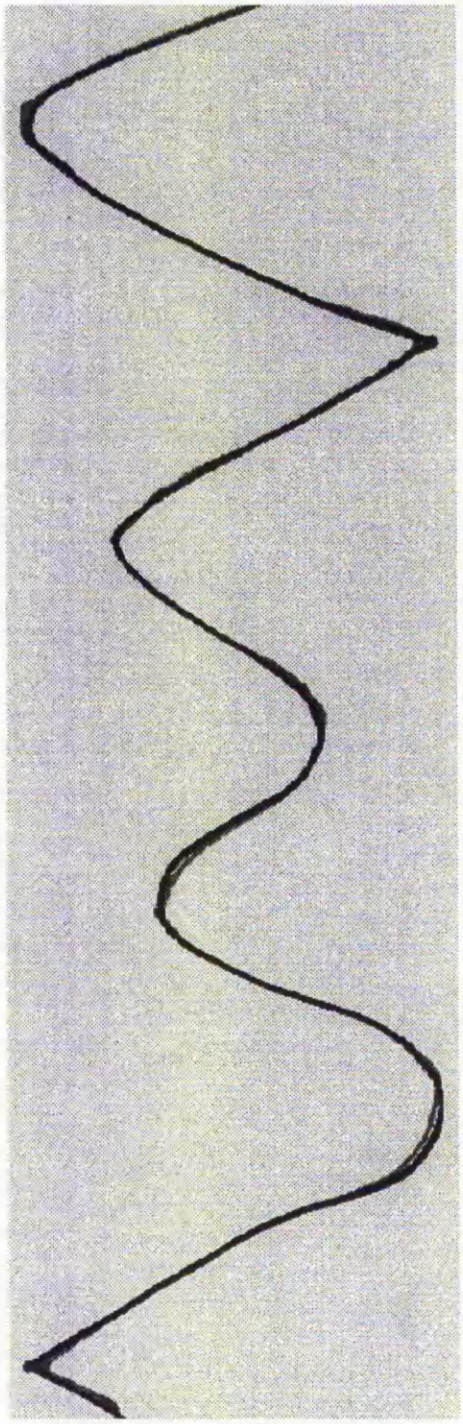


Figure 2V-D-4: Frame Three

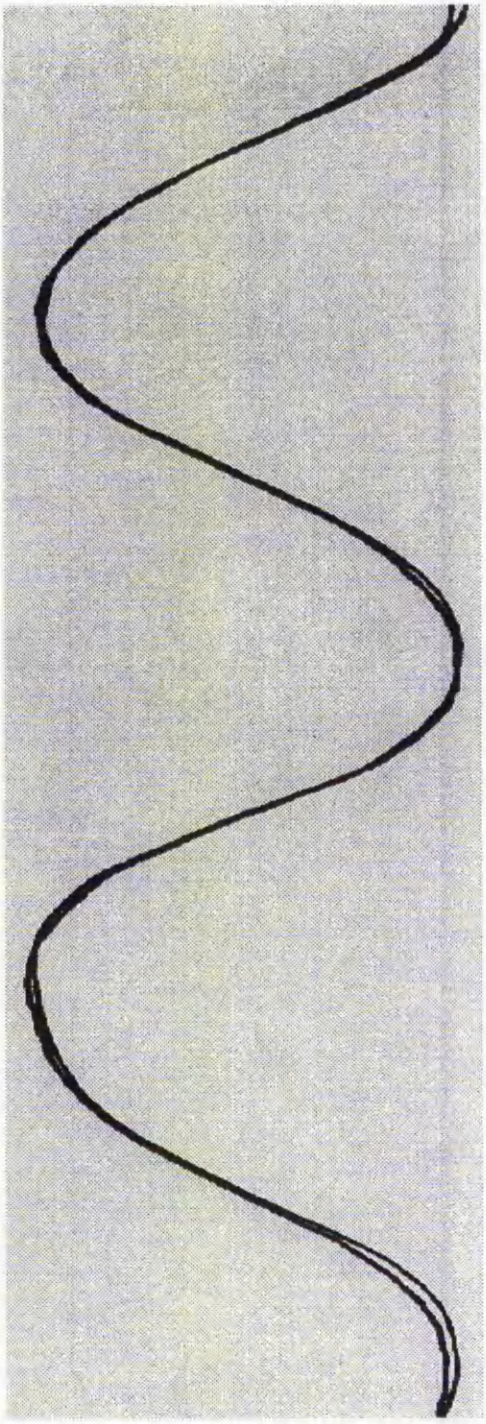


Figure 3V-1: Sample One (3VMethod)

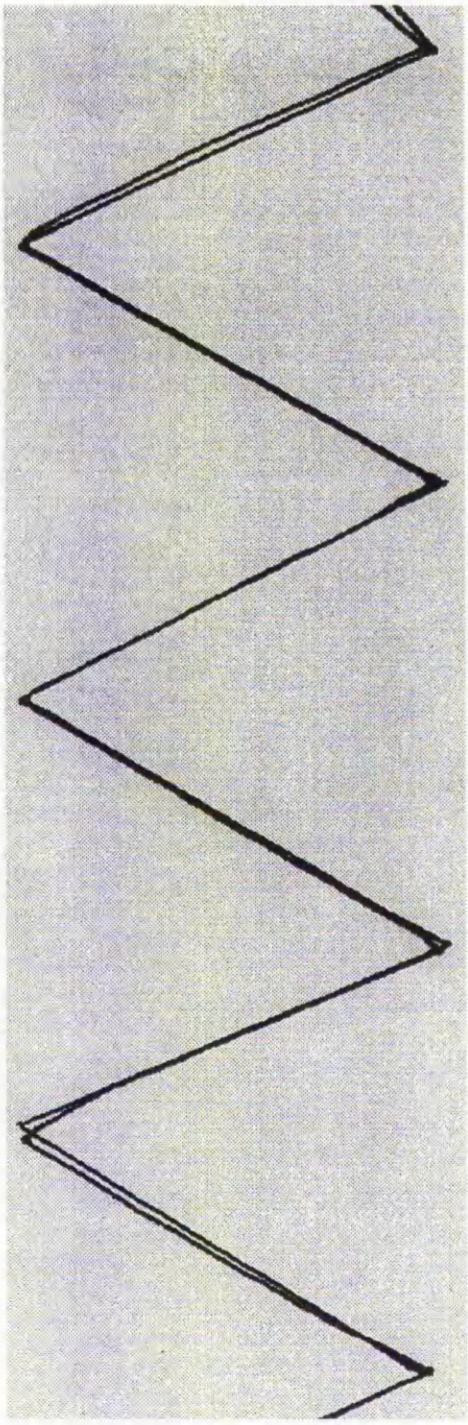


Figure 3V-2: Sample Two (3VMethod)

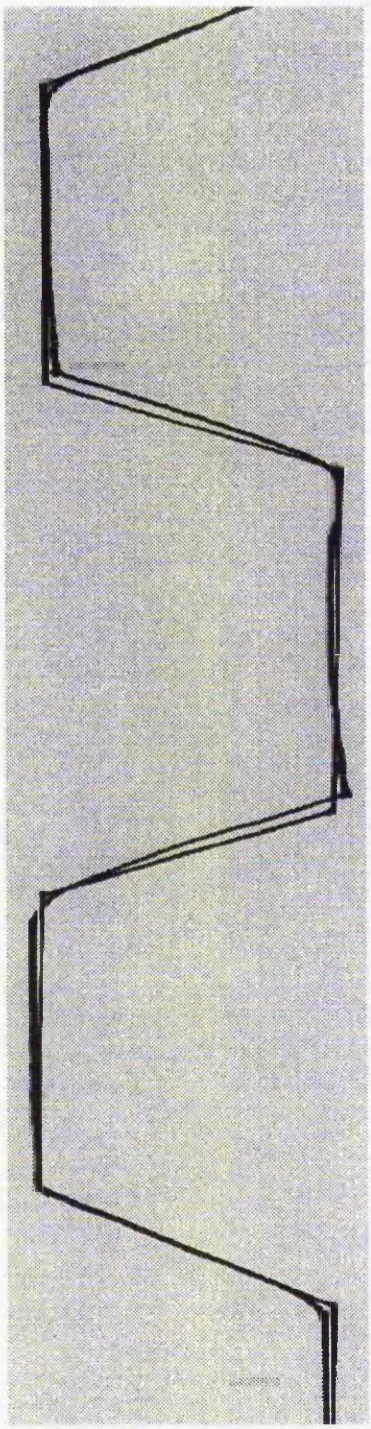


Figure 3V-3: Sample Three (3VMethod)

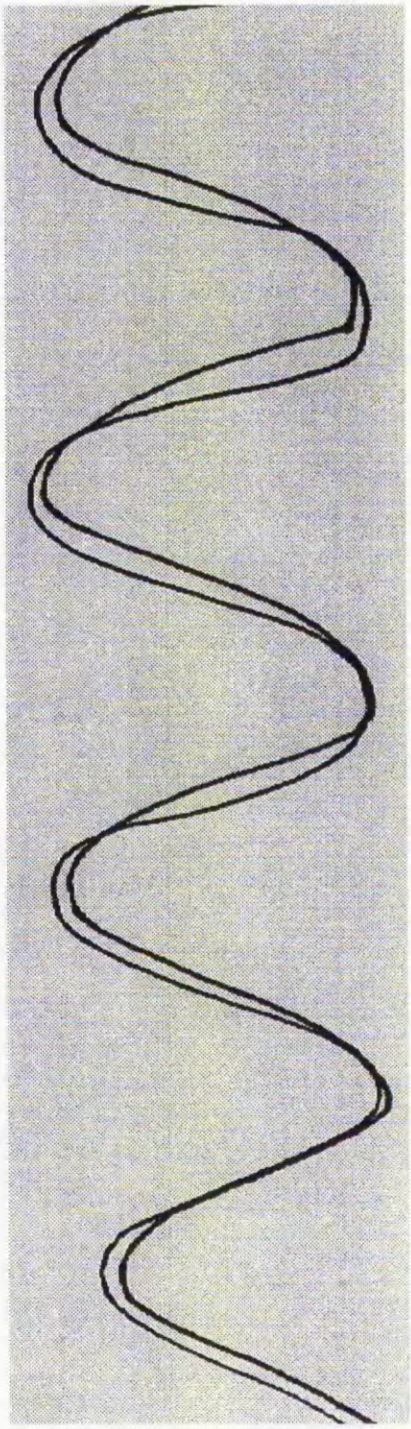


Figure 3V-4: Sample Four (3VMethod)

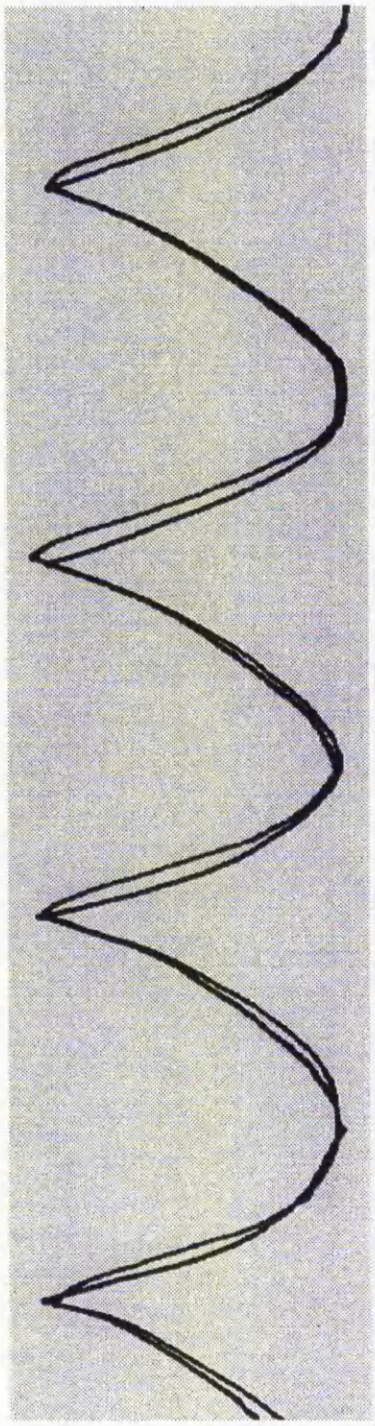


Figure 3V-5: Sample Five (3VMethod)

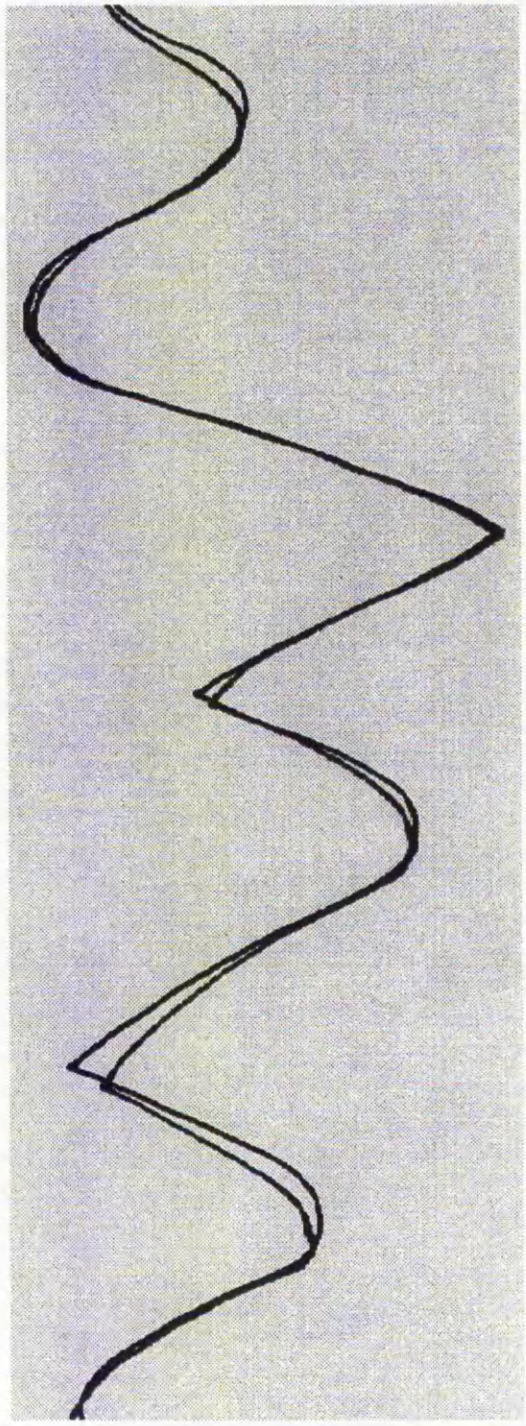


Figure3V-6: Sample Six (3VMethod)

Appendix D

SAMPLES OF SMP FOLLOWING USING THE GENERIC ERROR COMPENSATION ALGORITHM

This appendix provides various samples taken from the experiments of SMP (*Spring Mounted Pen*) following using the modified vision of 2VMethod in which the *GEC Algorithm (Generic Error Compensation Algorithm)* is applied. A number of regular and irregular shapes of curves are used in the experiments to verify the effectiveness of this learning algorithm.

In contrast to the standard 2VMethod with the PEC Algorithm (Piecewise Error Compensation Algorithm), the results of SMP following applied the GEC Algorithm indicate that the technique developed is only required one learning frame in order to successfully minimise the path-following-errors.

Figure D-1 to Figure D-5 illustrate five regular shapes of curves together with the results of SMP following. The original desired patterns and their compensated paths are also included. Figure D-6 to Figure D-10 depict five different irregular shapes of drawing patterns. The compensated paths created by the A.I. kernel using the inexact algorithms are illustrated.

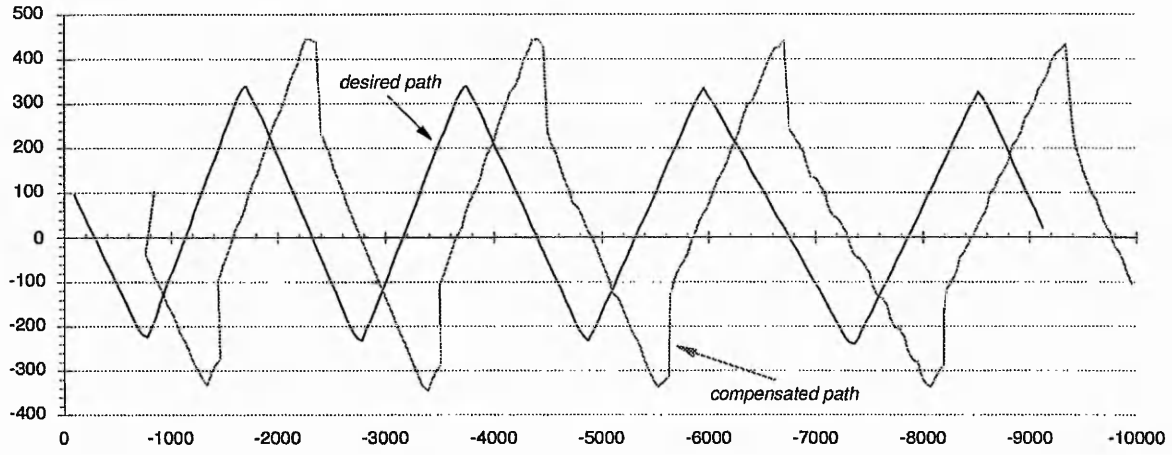
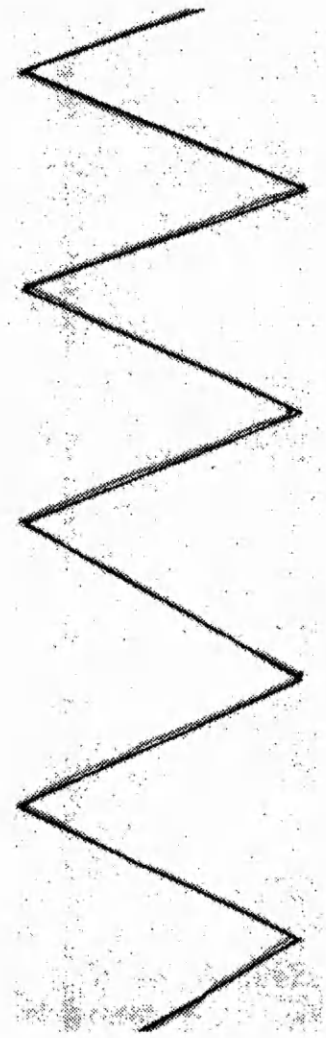

500
400
300
200
100
0
-100
-200
-300
-400
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000
desired path
compensated path

Figure D-1



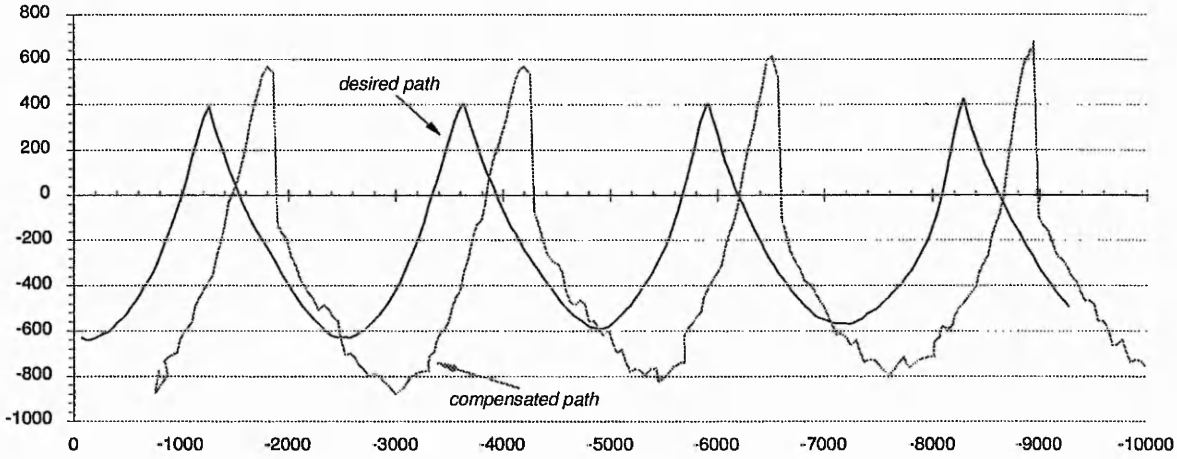
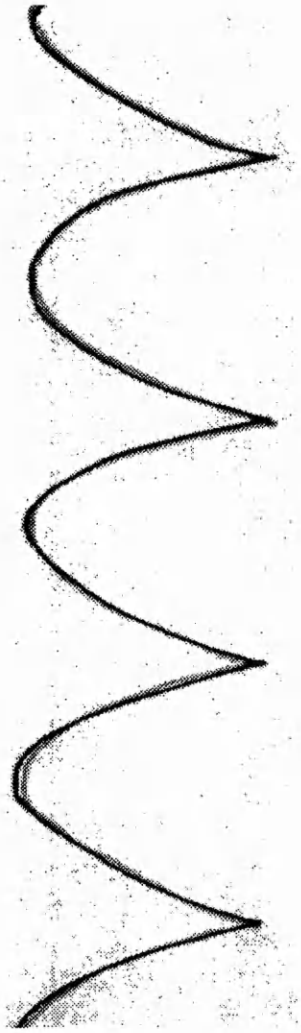

800
600
400
200
0
-200
-400
-600
-800
-1000
desired path
compensated path
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000

Figure D-2



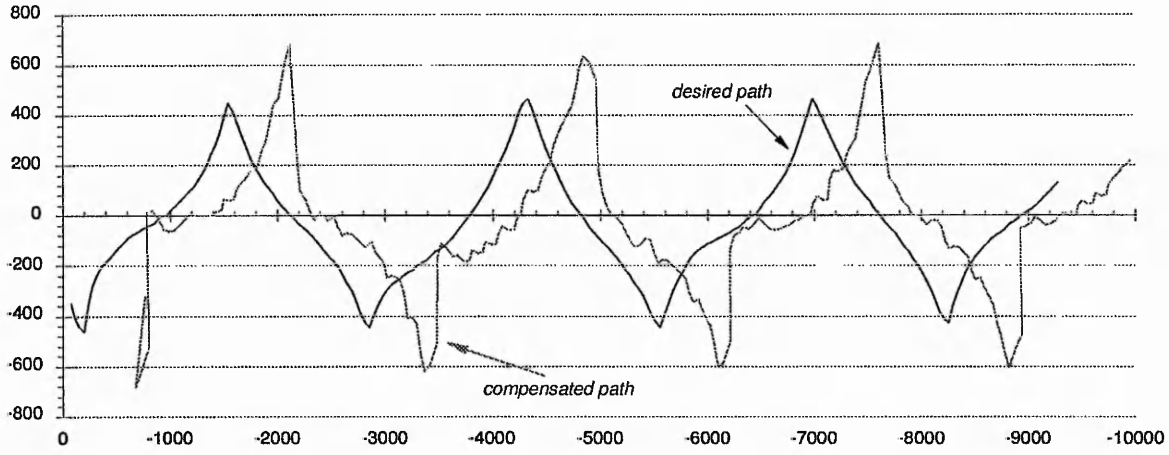

800
600
400
200
0
-200
-400
-600
-800
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000
desired path
compensated path

Figure D-3

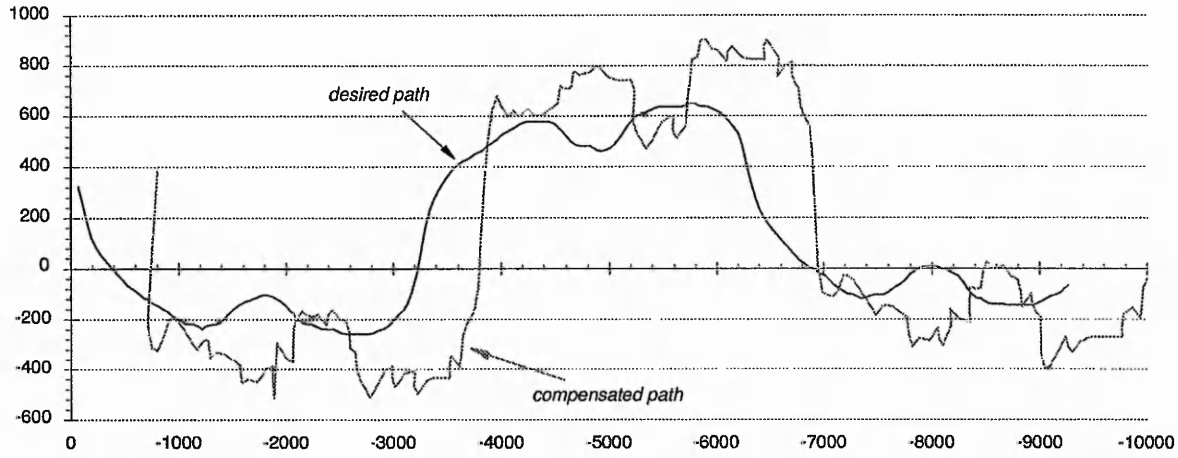

1000
800
600
400
200
0
-200
-400
-600
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000
desired path
compensated path

Figure D-4

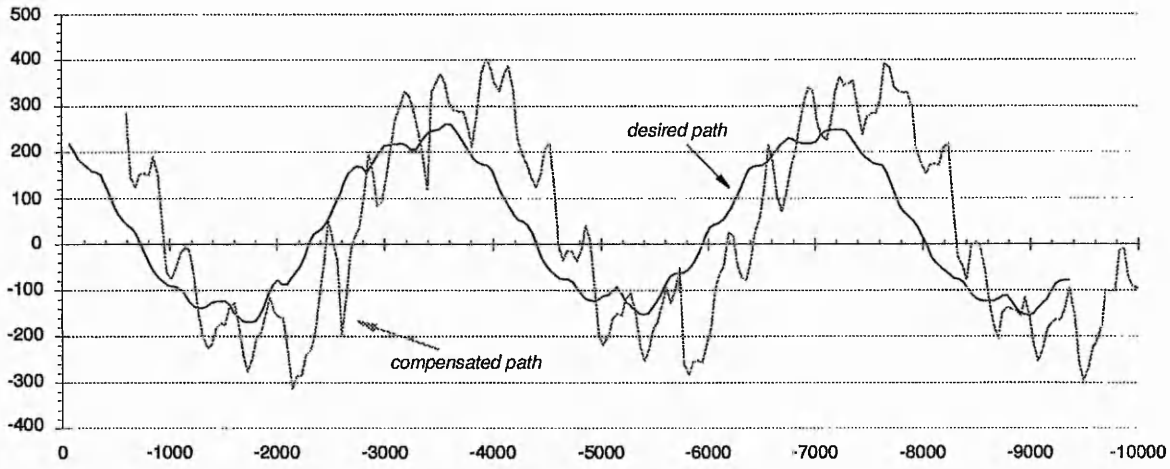

500
400
300
200
100
0
-100
-200
-300
-400
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000
desired path
compensated path

Figure D-5

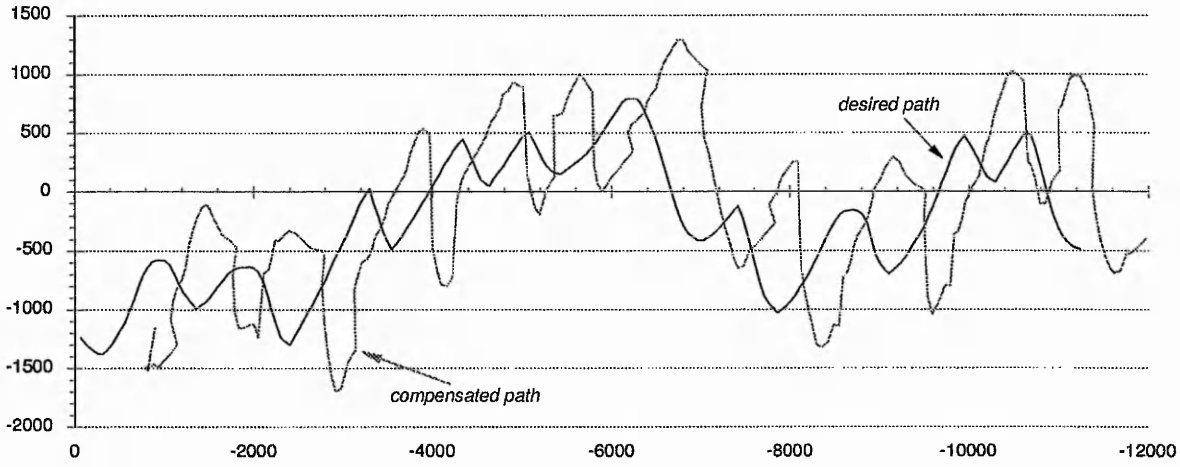

1500
1000
500
0
-500
-1000
-1500
-2000
0
-2000
-4000
-6000
-8000
-10000
-12000
desired path
compensated path

Figure D-6

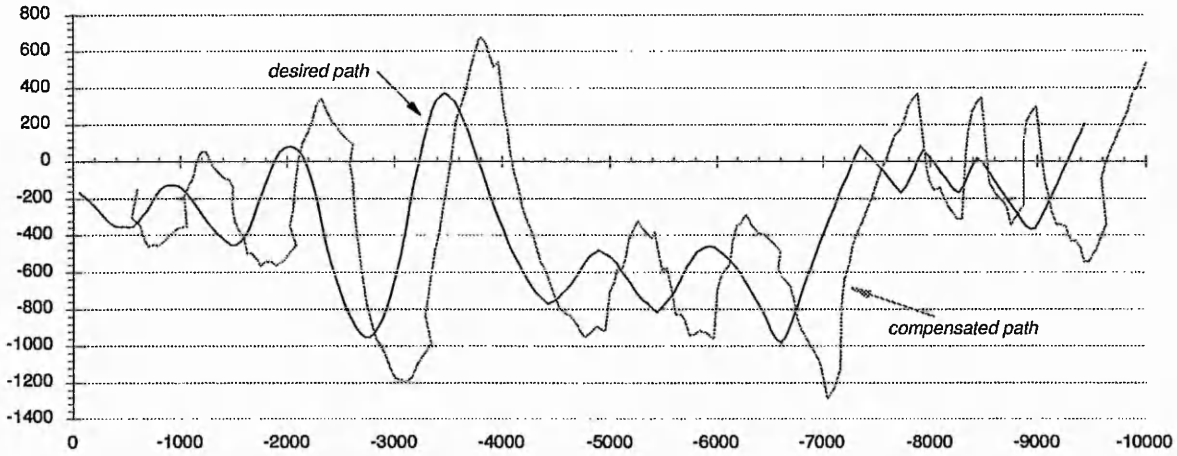

800
600
400
200
0
-200
-400
-600
-800
-1000
-1200
-1400
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000
desired path
compensated path

Figure D-7

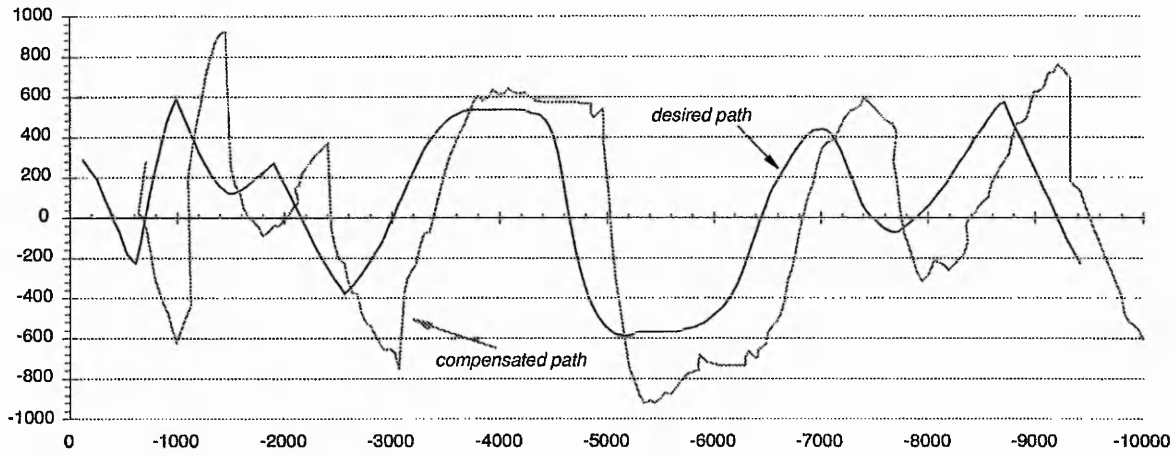
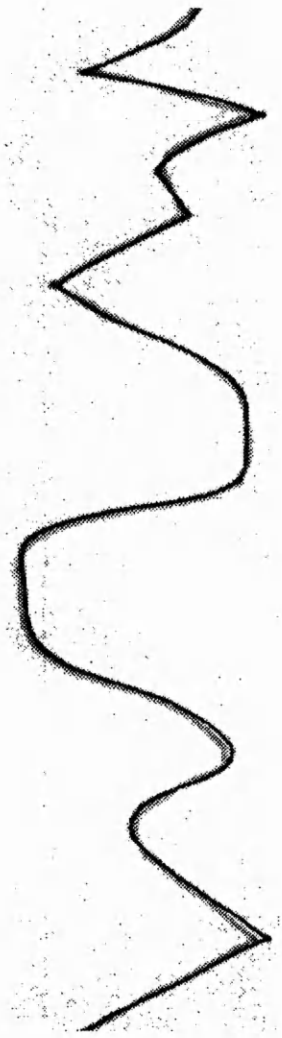

1000
800
600
400
200
0
-200
-400
-600
-800
-1000
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000
desired path
compensated path

Figure D-8



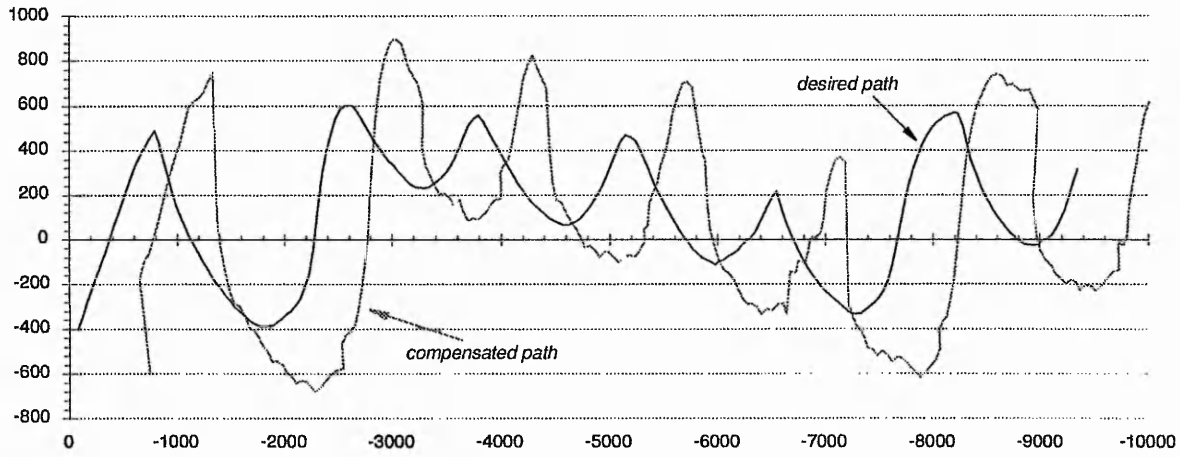

1000
800
600
400
200
0
-200
-400
-600
-800
0
-1000
-2000
-3000
-4000
-5000
-6000
-7000
-8000
-9000
-10000
desired path
compensated path

Figure D-9

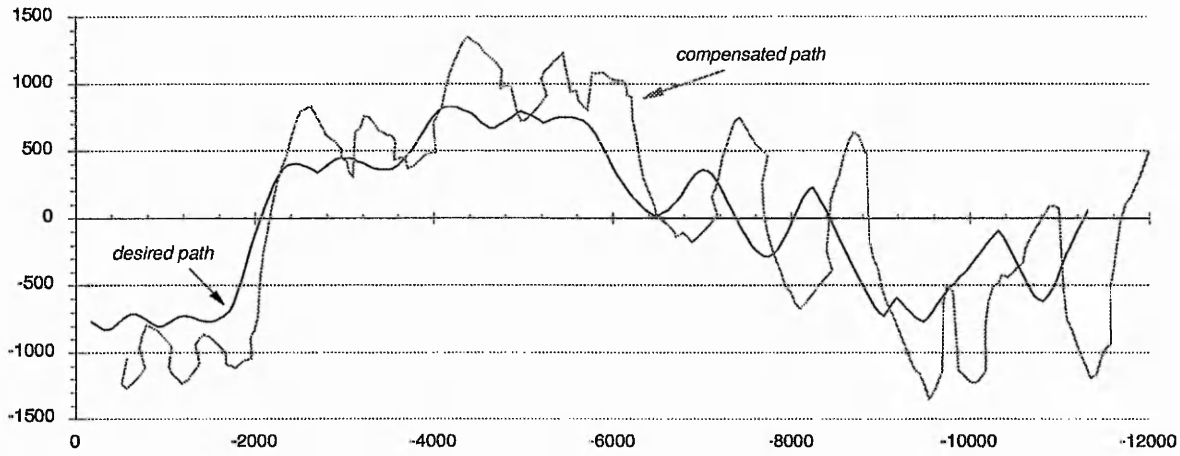

1500
1000
500
0
-500
-1000
-1500
0
-2000
-4000
-6000
-8000
-10000
-12000
compensated path
desired path

Figure D-10

Appendix E
PUBLISHED PAPERS

The work described in this document has been published by the author in a number of papers. References are provided together with the tag used in the document text. Full copies of the papers are also enclosed.

[SHE94b]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Applications. Kawasaki, Japan, 1994.

[SHE95a]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology. California, USA, 1995.

[SHE95b]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "Tightly coupled vision based control system using inexact algorithms", IEEE Second Asian Conference on Computer Vision. Singapore, 1995.

[SHE96a]

Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A Vision Based Control System Using Neural Fuzzy Theory", Fourth Iranian Conference on Electrical Engineering. ICEE-96. Tehran, Iran, 1996.

[SHI95a]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE Fifth International Conference on Image Processing and Its Applications. Edinburgh, U.K., 1995.

[SHI95b]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Automation of lace cutting using real-time vision", 8th International Congress on Condition Monitoring and Diagnostic Engineering Management. Queen's University, Canada, 1995.

[SHI96a]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Close coupling of pre- and post-processing vision stations using Inexact Algorithm", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology, California, USA, 1996.

[SHI96b]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Correction of errors due to flexibility of dynamic systems", 1996 IEEE International Conference on Robotics and Automation. Minnesota, U.S.A., 1996.

[SHI96c]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "An automatic lace trimming process using real-time vision", Journal of Real-Time Imaging, April, 1996.

[SHI96d]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Inexact Algorithm for correction of errors due to flexibility of dynamic structures", IEEE 8th Mediterranean Electrotechnical Conference - Industrial Applications in Power Systems, Computer Science and Telecommunications. Bari, Italy, 1996.

[SHI96e]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Neural fuzzy based self-learning algorithms for handling flexibility of dynamic structures", The 22nd Annual International Conference of the IEEE Industrial Electronics Society. Taipei, Taiwan, 1996. *(nominated for the IEEE Prize Paper Award)*

[SHI96f]

Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Generic error compensation algorithm for managing flexibility of dynamic structures using neural fuzzy approaches", 1996 IEEE International Conference on Systems, Man and Cybernetics. Beijing, China, 1996. *(to be published)*

A FUZZY REASONING RULE-BASED SYSTEM FOR LACE PATTERN DETECTION

Nasser Sherkat, Chi-Hsien V. Shih, Peter Thomas

The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom
Tel: +44 105 9486538 Fax: +44 105 9486518

ABSTRACT

Lace is liable to stretch as it is passed through the feed mechanism, in which a vision system is engaged to detect the changes of the motif and find the cutting path (river) across the lace pattern. The vision system has to work with many different lace patterns, sizes and tolerate misalignment, stretch and other distortions. A *Fuzzy Reasoning Rule-based technique* is employed in order to overcome the problems of flexibility. Several experiments have been carried out using lace patterns of varying complexity. All cutting paths across the patterns were successfully found. Experimental results indicate that this method can correctly detect the river path in different lace patterns.

I. INTRODUCTION

Handling lace in terms of cutting it along the designed paths is usually carried out manually. Skilled operators use high speed rotating blades or hot wire to cut the lace along the designated path. In order to satisfy industrial requirements two main conditions must be satisfied [4]. Firstly, to achieve a sufficient degree of automation the river must be found without prior knowledge of the lace pattern. Secondly, the process of river location must be carried out in real-time. To achieve this, the extracted knowledge can be used to speed up the search for the river in subsequent frames. The resolution required for image analysis is considerably lower than that provided by general purpose Charge Coupled Device camera (Figure 1). A bi-level image merely consisting of bright and dark areas would suffice (Figure 2) [4].

In white or near white lace, after the thresholding operation, a river shows up as a dark area (pixel group) within the edges that crosses from one side of the image to other in a nearly unbroken sequence (Figure 2). There are *thick white threads* that cross the river at intervals that are indistinguishable from the material surrounding the river (marked by circles in Figure 2). Allowance must be made for small breaks in continuity of the river due to

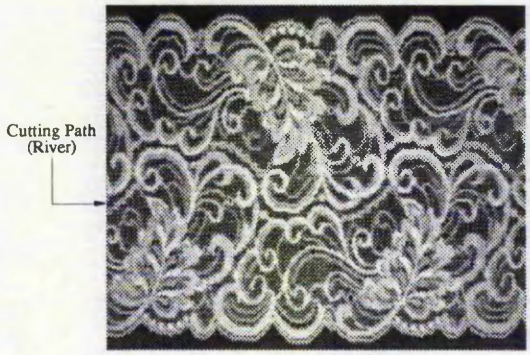


Figure 1: *Lace image received from the CCD camera*

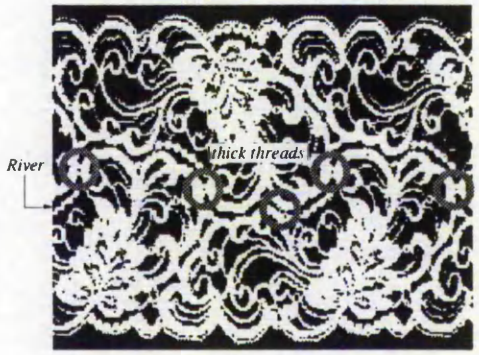


Figure 2: *Bi-level lace bitmap image*

these cross threads.

Lace comprises a fine and intricate pattern, with various densities of knit and holes. On most designs the mirrored pattern repeats many times, but in practice the repeats are never absolutely identical. Furthermore, lace is flexible, extensible and easily distorts, effectively changing the pattern. Norton-Wayne [1] experienced this problem and states this characteristic of lace making it impossible to cut in a consistent position. Russell [3] *et al.* approached this problem by trying to locate a reference feature in the lace motif so they can keep track of the change in the pattern due to stretch. Moreover, the vision system has to work with many different lace patterns and sizes and tolerate misalignment, stretch and other distortions [4]. To overcome the flexibility problem, we employ an inexact decision making theory - fuzzy rule-based inference technique.

In classical normative decision theory the components of the basic model of decision making under certainty are taken to be crisp sets or functions. By *crisp* we mean dichotomous - that is, of the yes-or-no type rather than of the more-or-less type" [8]. The set of actions is as precisely defined as the set of possible states and the utility function is also assumed to be precise. In descriptive decision theory this precision is no longer assumed; but ambiguity and vagueness are very often modeled verbally, which usually does not permit the use of powerful mathematical methods for purposes of analysis and computation. The presented approach draws on this characteristic to cope with the flexibility problems described above.

II. PATTERN RECOGNITION

The scheme for applying fuzzy inference techniques to find the first river across the lace pattern with no previous knowledge can be broken down into the following tasks:

- Defining system input and output membership functions;
- Fuzzification process;
- Inference and composition;
- Defuzzification process;
- Verification.

The system reads two input variables (position and density) after each black pixel group has been processed. The fuzzification process then assigns a value to represent an input's degree of membership in one or more fuzzy sets. During inference and composition process, strengths are computed based on antecedent values and then assigned to the rules' fuzzy output. Finally, the defuzzification process employs compromising techniques to calculate the average weight for system output. These steps are described in detail as follows.

2.1 Defining system input and output membership functions

The degree of membership is decided from overlapping sets of a membership function, which is defined normally based on intuition or experience. The pre-defined membership functions cover the entire range of values for system input and output, and will define a degree of truth for every point in the universe of discourse. The shapes and number of fuzzy-set membership functions we chose depend on parameters such as the required exactitude, steadiness and responsiveness of the system. Different shapes such as triangles and trapezoids are often employed to define fuzzy-set membership functions [7][8].

The objective here is to find the river along a lace pattern, by using linguistic variables to

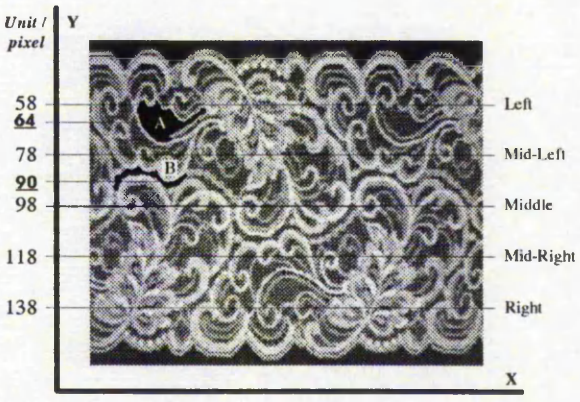


***Figure 3**: Corresponding positions for black pixel group A and B*

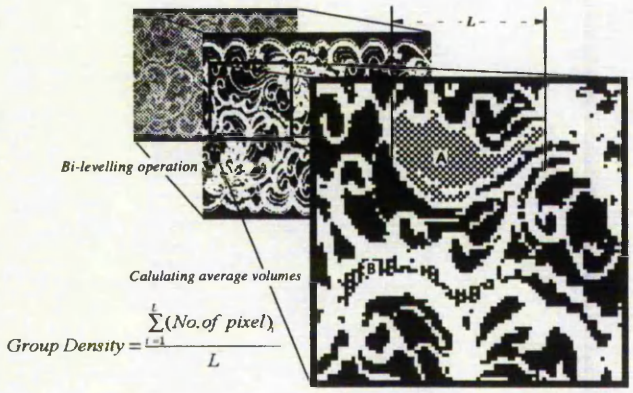


Figure 4: *A example for calculating the group densities for group A and B*

represent the common feature of the river shape in various lace patterns. These common features may be described as:

i) that the *position* of the river is around the *centre* of a lace pattern;
ii) the river pixel group density is not large.

From these linguistic descriptions, two system inputs, *group position* and *group density*, can be defined. By monitoring the position and density of the black pixel groups (Figure 2) across a lace pattern, a fuzzy decision making system can determine whether the pixel group is a possible segment of the river. Figure 3 and Figure 4 illustrate the two system inputs corresponding to an example lace pattern together with two candidate groups A and B.

Two initial experiments were carried out to define the system input and output membership functions. Frequency histograms were used on the sample data to define input membership functions [2][5]. From these experimental results we can obtain a set of data from the *River group* part to define the membership functions. The triangular membership function is most common and has proved to be a good compromise between effectiveness and efficiency. Overlapping between fuzzy-set boundaries is desirable and the key to smooth operation of the system. To simplify the procedure of defining fuzzy membership functions, an overlap of 50 percent between adjacent fuzzy sets is used in this experiment. Besides, each fuzzy set is chosen according to the central value and the slope on either side (Figure 5).

2.2 Fuzzification process

Fuzzification is the procedure of calculating an input value to represent a degree of membership in one or more fuzzy sets. This process uses two basic steps which are repeated for each system input. First, a crisp input has to be read and scaled to a value between 0 and 100. Second, the input must be translated to a degree of membership function. Figure 5 shows two system inputs, *position* and *density*. Each value of system input has a degree of membership in each of these sets. Once the degrees of memberships are assigned, the values are used to evaluate the rules.

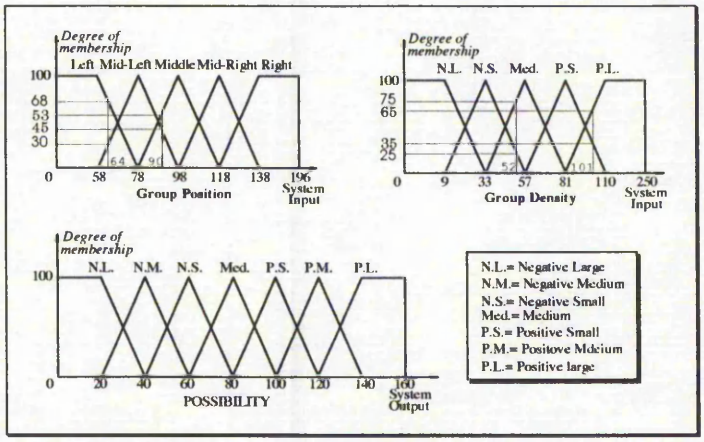


Figure 5: *System input and output membership functions*

Rule 1:	IF	position is Left	AND density is N.L.	THEN	possibility is N.M.
Rule 2:	IF	position is Left	AND density is N.S.	THEN	possibility is P.S.
Rule 3:	IF	position is Left	AND density is Med.	THEN	possibility is P.S.
Rule 4:	IF	position is Left	AND density is P.S.	THEN	possibility is N.M.
Rule 5:	IF	position is Left	AND density is P.L.	THEN	possibility is N.L.
Rule 6:	IF	position is Mid-Left	AND density is N.L.	THEN	possibility is P.S.
Rule 7:	IF	position is Mid-Left	AND density is N.S.	THEN	possibility is P.M.
Rule 8:	IF	position is Mid-Left	AND density is Med.	THEN	possibility is P.L.
Rule 9:	IF	position is Mid-Left	AND density is P.S.	THEN	possibility is Med.
Rule 10:	IF	position is Mid-Left	AND density is P.L.	THEN	possibility is N.L.
Rule 11:	IF	position is Middle	AND density is N.L.	THEN	possibility is P.S.
Rule 12:	IF	position is Middle	AND density is N.S.	THEN	possibility is P.M.
Rule 13:	IF	position is Middle	AND density is Med.	THEN	possibility is P.L.
Rule 14:	IF	position is Middle	AND density is P.S.	THEN	possibility is P.S.
Rule 15:	IF	position is Middle	AND density is P.L.	THEN	possibility is N.L.
Rule 16:	IF	position is Mid-Right	AND density is N.L.	THEN	possibility is P.S.
Rule 17:	IF	position is Mid-Right	AND density is N.S.	THEN	possibility is P.M.
Rule 18:	IF	position is Mid-Right	AND density is Med.	THEN	possibility is P.L.
Rule 19:	IF	position is Mid-Right	AND density is P.S.	THEN	possibility is Med.
Rule 20:	IF	position is Mid-Right	AND density is P.L.	THEN	possibility is N.L.
Rule 21:	IF	position is Right	AND density is N.L.	THEN	possibility is N.M.
Rule 22:	IF	position is Right	AND density is N.S.	THEN	possibility is P.S.
Rule 23:	IF	position is Right	AND density is Med.	THEN	possibility is P.S.
Rule 24:	IF	position is Right	AND density is P.S.	THEN	possibility is N.M.
Rule 25:	IF	position is Right	AND density is P.L.	THEN	possibility is N.L.

Figure 6: *System rule base*

2.3 Inference and composition

Fuzzified inputs are processed through a pre-defined set of rules using min-max evaluation to form fuzzified outputs. The author developed a set of rules that have the form of

IF [*antecedent_1*] AND [*antecedent_2*]
THEN [*consequence*]

which are listed in Figure 6. The antecedents of rules correspond directly to degrees of membership calculated during the fuzzification process. Each antecedent has a degree of truth assigned to it as a result of fuzzification.

In inference and composition processes, strengths are enumerated based on antecedent values and then assigned to the rules' output strengths. Figure 7 illustrates the actual fuzzy outputs calculated during rule evaluation process for pixel group A. The strength of a rule is assigned the value of the weakest (*minimum*) antecedent. As more than one rule applies to the same specific action, the strongest (*maximum*) value of rules is used :

i) from Rule 4:

$N.M._{rule\ strength}$
= min (*antecedent_1, antecedent_2*)
= min (68, 35) = 35

ii) Rule 5:

$N.L._{rule\ strength1}$ = min (68, 65) = 65,
from Rule 10 also
$N.L._{rule\ strength2}$
= min (30, 65) = 30
then the maximum rule strength on fuzzy set N.L. is

$N.L._{rule\ strength}$
= max (65, 30) = 65

Rule1:	IF position is 68	AND	density is 0	THEN possibility is N.M.
Rule2:	IF position is 68	AND	density is 0	THEN possibility is P.S.
Rule3:	IF position is 68	AND	density is 0	THEN possibility is P.S.
Rule4:	IF *position is 68*	AND	*density is 35*	THEN possibility is *N.M.*
Rule5:	IF *position is 68*	AND	*density is 65*	THEN possibility is *N.L.*
Rule6:	IF position is 30	AND	density is 0	THEN possibility is P.S.
Rule7:	IF position is 30	AND	density is 0	THEN possibility is P.M.
Rule8:	IF position is 30	AND	density is 0	THEN possibility is P.L.
Rule9:	IF *position is 30*	AND	*density is 35*	THEN possibility is *Med.*
Rule10:	IF *position is 30*	AND	*density is 65*	THEN possibility is *N.L.*
Rule11:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule N:	*IF position is A*	*AND*	*density is B*	*THEN possibility is C.....*

Figure 7: *Inference and composition for pixel group A*

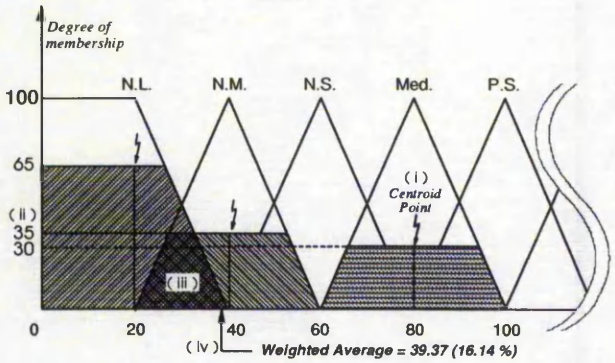


Figure 8: *Defuzzification process for pixel group A*

iii) Rule 9:

$Med.^{rule\ strength}$

$= \min(30, 35) = \underline{30}$

2.4 Defuzzification process

Defuzzification process is to convert its fuzzy outputs into a signal raw or crisp output. There are many defuzzification methods. In these experiments, we chose the *"centre-of-gravity method"* which is a common and accurate defuzzification technique for resolving both the vagueness and conflict [6][7]. Figure 8 illustrates defuzzification of the output using the centre of gravity method. The weighted average is calculated as follows:

$$Weighted\ average = \frac{\sum(shaded\ area \times centroid\ point)}{\sum(shaded\ area)}$$

By relying on the use of fuzzy inference technique, each black pixel group is calculated and assigned an average weight (*possibility*). For instance, in Figure 4, the output value for group A is 39.37 (*16.14 %*) (see Figure 8), also group B is 134.64 (*95.53 %*). Since the average weight of group A is only 16.14% (less than 50%), the pixel group only has a 16 percent *possibility* of being a segment of the river. Therefore group A is not a part of the river.

2.5 Verification

Once *all* the black pixel groups have been assigned a *possibility value* (average weight), pixel groups whose possibility values are less than 80 (50%) are abandoned (Figure 9). The verification process can then be broken down into the following tasks:

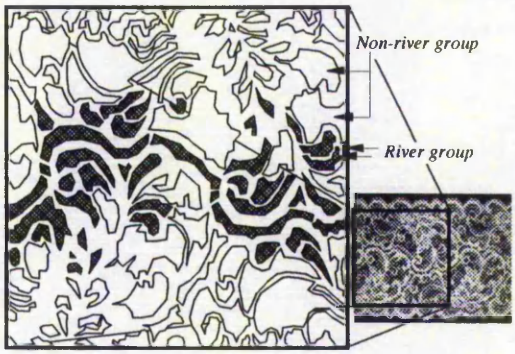


Figure 9: *Each possible river segments whose weights are bigger than 80 (50%)*

i) Calculate the distance between two adjacent groups;
ii) If the distance is shorter than a specified value (set to six pixels long in these experiments) a network is built to record this path;
iii) Continuously trace the distances between pixel groups while recording all the correct paths until a new pixel group reaches the border of the image (right hand edge of the frame);
iv) Calculate the total possibility values and divide by the number of the group in this path (*average possibility*);
v) If the *average possibility* is bigger than a specified value, (110 or 75% was used in the experiments) then the correct river has been found; if the average possibility is less than this value, repeat step (iii) to (v) until the correct river is located.

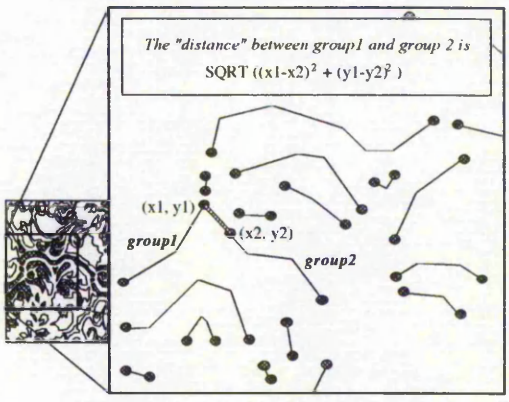


Figure 10 : *An example for calculating the distance between pixel groups*

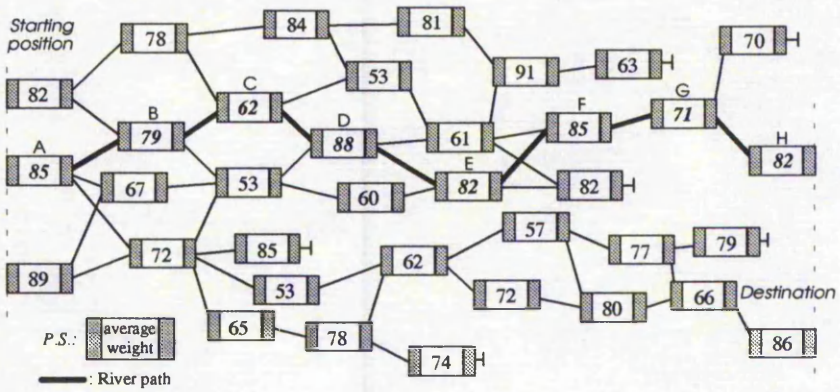


Figure 11: *Interconnection between each possible river segments*

Figure 10 illustrates the computation of the distance between two adjacent pixel groups. By calculating the distances and tracing the average possibilities in all these segments, the river location, highlighted in Figure 11, can be pin-pointed.

III. EXPERIMENTAL RESULTS

A number of experiments were carried out to evaluate the effectiveness of this method. Various kinds of lace patterns were employed for detecting the river location. All cutting paths across the patterns were successfully found. The time taken to isolate the river and produce cutting path depends on complexity of the pattern. Time taken for most kinds of motif is typically about 0.3 second using an Intel 80486 processor running at 66 MHz. However, in the case of a very few intricate lace patterns (e.g. Figure 1), up to 1.5 seconds is required. Once the river path on the first frame is found, this knowledge can be utilised to speed up the detection for the river in subsequent frames to meet the real-time requirements of the system. Some sample laces together with the resulting river path are shown in Figure 12 and Figure 13.

IV. CONCLUSION

We have described attempts to develop a fuzzy reasoning rule-based system for detecting various kinds of lace patterns. Experimental results indicate that the objectives have mostly been fulfilled. The system requires no prior knowledge of any particular lace pattern or training. According to the results this method can precisely detect the proper river path within diversified lace patterns. Comparing with the previously reported methods [3][4], it is not only relatively easy to design and implement the system by means of using the fuzzy reasoning techniques, but also more maintainable.

Figure 12: *Example A of river extraction*

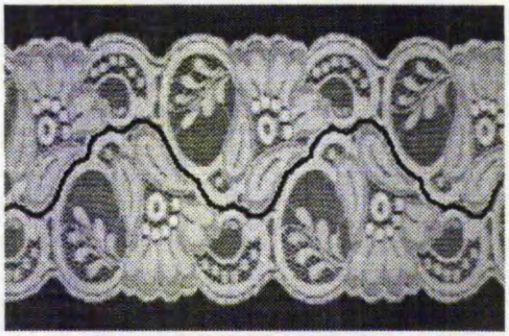


Figure 13: *Example B of river extraction*

REFERENCES

[1] Norton-Wayne, L. "Inspection of lace using machine vision", Computer Graphics Forum, vol. 10, iss.2, June 1991, pp 113 - 19.

[2] Roberts, D.W. "Analysis of forest succession with fuzzy graph theory", Ecological Modeling, 45:261-274, 1989.

[3] Russell, R.A. and Wong, P. "Automation of lace cutting using computer vision", Robots: Coming of Age. Proceedings of International Symposium and Exposition on Robots. Designed the 19th ISIR by the International Federation of Robotics, Nov. 1988, pp 385-93.

[4] Sherkat, Nasser; Birch, Mike and Thomas, Peter. "Real-time vision for automatic lace cutting", IS&T/SPIE Symposium on Electronic Imaging Science & Technology, 6-10 February 1994.

[5] Turksen, I.B. "Measurement of fuzziness: interpretation of the axioms of measure", Proceeding of the Conference on Fuzzy Information and Knowledge Representation for Decision Analysis, IFAC, Oxford, 1984, pp 97-102.

[6] Viot, Greg. "Fuzzy logic in C : creating a fuzzy-based inference engine", Dr. Dobb's Journal, February 1993, pp 40 - 49 & 94.

[7] Zadeh, Lofti A. "Fuzzy logic", Computer, IEEE, April 1988, pp 83 - 93.

[8] Zimmermann, H. J. Fuzzy Sets, Decision Making, and Expert Systems, Kluwer Academic Publishers, Lancaster, 1987.

A fuzzy reasoning rule-based system for handling lace pattern distortion

Nasser Sherkat, Chi-Hsien V. Shih, Peter Thomas

The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom

ABSTRACT

Much progress has been made in using computer vision to automate the process of lace scalloping. Because the material is flexible, dealing with stretch due to mechanical feed produces a challenge. The vision system has to work with many different patterns and sizes of lace as well as tolerating misalignment. A *Fuzzy Reasoning Rule-based technique* is employed in order to overcome the problems of material flexibility. Several experiments have been carried out using lace patterns of varying complexity. All cutting paths across the patterns were successfully found. Experimental results indicate that this method can correctly detect the river path in different lace patterns, and cope with lace stretch as well as distortion.

Keywords: lace pattern, fuzzy reasoning, image analysis, CCD camera, FAM bank, vision system

1. INTRODUCTION

Handling lace in terms of cutting the material along the designed paths is usually carried out manually. Skilled operators use high speed rotating blades or hot wire to cut the lace along the designated path which is illustrated in Figure 1. In order to satisfy industrial requirements two main conditions must be satisfied.[5] Firstly, to achieve a sufficient degree of automation the river must be found without prior knowledge of the lace pattern. Secondly, the processes of river location and cutting must be carried out in real-time. To achieve this, the extracted knowledge can be used to speed up the search for the river in subsequent frames. The resolution required for image analysis is considerably lower than that provided by a general purpose Charge Coupled Device camera (Figure 2). A bi-level image merely consisting of bright and dark areas would suffice (Figure 3).[5]

In white or near white lace, after the thresholding operation, a river shows up as a dark area (pixel group) within the edges that cross from one side of the image to the other in a nearly unbroken sequence (Figure 3). There are *thick threads* that cross the river at intervals that are indistinguishable from the material surrounding the river (marked by circles in Figure 3). Allowance must be made for small breaks in continuity of the river due to these cross threads.

Figure 1: *A typical lace pattern*

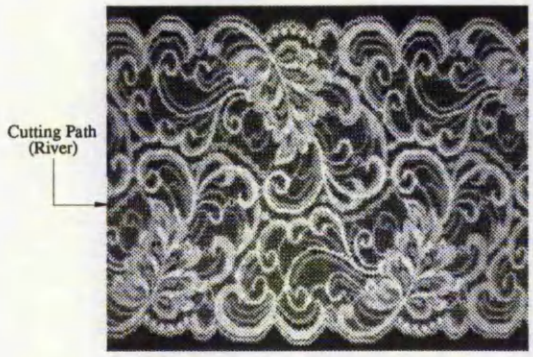


Figure 2: *Lace image received from the CCD camera*

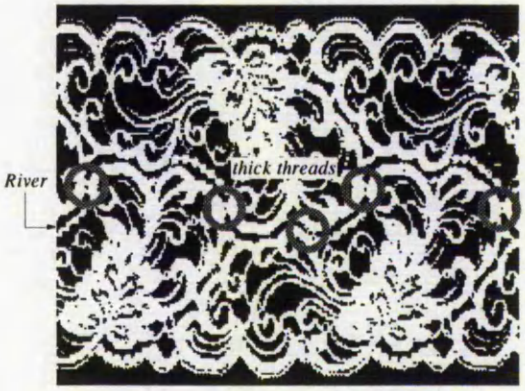


Figure 3: *Bi-level lace bitmap image*

Lace comprises a fine and intricate pattern, with various densities of knit and holes. On most designs the mirrored pattern repeats many times, but in practice the repeats are never absolutely identical. Furthermore, lace is flexible, extensible and easily distorts, effectively changing the pattern. Norton-Wayne[2] experienced this problem and states that this characteristic of lace makes it impossible to cut in a consistent position. Russell[4] *et al.* approached this problem by trying to locate a reference feature in the lace motif so they can keep track of the change in the pattern due to stretch. Moreover, the vision system has to work with many different lace patterns and sizes and tolerate misalignment, stretch and other distortions.[5] To overcome the flexibility problem, we employ an inexact decision making theory based on fuzzy rule-based inference technique.

In classical normative decision theory the components of the basic model of decision making under certainty are taken to be crisp sets or functions. By *crisp* we mean dichotomous - that is, of the yes-or-no type rather than of the more-or-less type.[9] The set of actions is as precisely defined as the set of possible states and the utility function is also assumed to be precise. In descriptive decision theory this precision is no longer assumed; but ambiguity and vagueness are very often modeled verbally, which usually does not permit the use of powerful mathematical methods for purposes of analysis and computation. The presented approach draws on this characteristic to cope with the flexibility problems described above.

2. PATTERN RECOGNITION

The scheme for applying fuzzy inference techniques to find the first river across the lace pattern with no previous knowledge can be broken down into the following tasks:

- Defining system input and output membership functions;
- Fuzzification process;
- Inference and composition;
- Defuzzification process;
- Verification.

Figure 4 illustrates the context data flow diagram of the system. This system reads two input variables *(Group Position and Density)* after each black pixel group has been processed. The fuzzification process then assigns a value to represent an input's degree of membership in one or more fuzzy sets. During inference and composition process, strengths are computed based on antecedent values and then assigned to the rules' fuzzy output. Finally, the defuzzification process employs compromising techniques to calculate the average weight for system output (Figure 5). These steps are described in detail as follows.

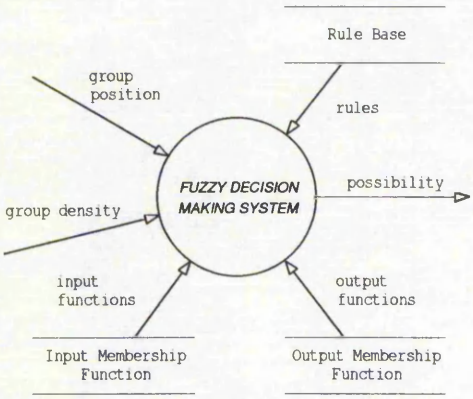


Figure 4: *Context diagram for system overview*

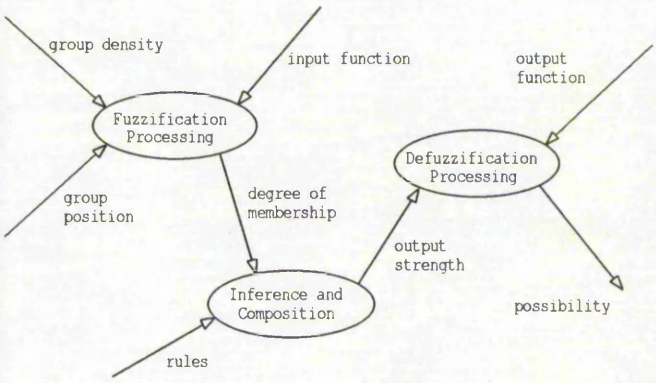


Figure 5: *Second level DFD for decision making process*

2.1 Defining system input and output membership functions

The degree of membership is decided from overlapping sets of a membership function, which is defined normally based on intuition or experience. The pre-defined membership functions cover the entire range of values for system input and output, and will define a degree of truth for every point in the universe of discourse. As the system is tuned to accomplish desired responses to given inputs or output, it is accepted that membership functions change several times. Nevertheless, once the system is in operation, these membership functions will not be modified. The shapes and number of fuzzy-set membership functions we choose depend on parameters such as the required exactitude, steadiness and responsiveness of the system. Different shapes such as triangles and trapezoids are often employed to define fuzzy-set membership functions[8,9].

The objective here is to find the river along a lace pattern, by using linguistic variables to represent the common feature of the river shape in various lace patterns. These common features may be described as:

i) the *position* of the river is around the *centre* of the pattern;
ii) the river pixel group density is not large.

From these linguistic descriptions, two system inputs, *group position* and *group density*, can be defined. By monitoring the position and density of the black pixel groups (Figure 3) across a lace pattern, a fuzzy decision making system can

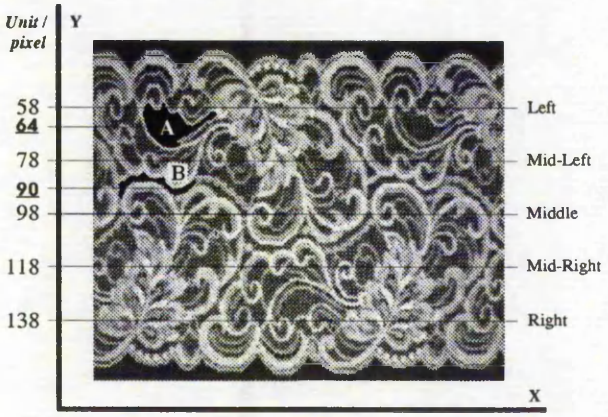


Figure 6*: Corresponding positions for black pixel group A and B*

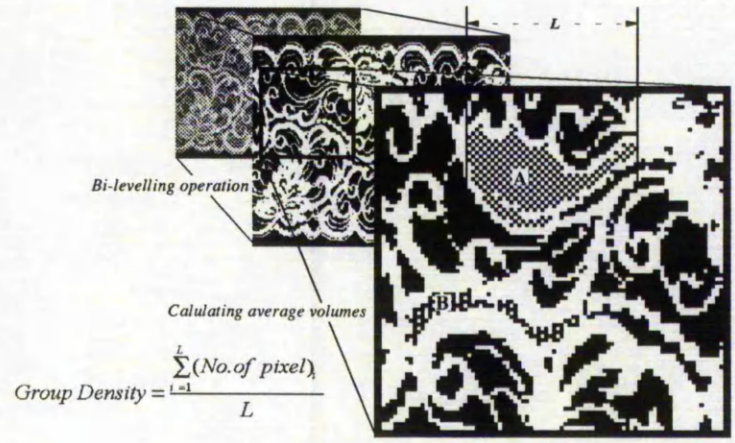


Figure 7: *A example for calculating the group densities for group A and B*

determine whether the pixel group is a possible segment of the river. Figure 6 and Figure 7 illustrate the two system inputs corresponding to an example lace pattern together with two candidate groups A and B.

Two initial experiments were carried out to define the system input and output membership functions. Figure 8 and Figure 9 illustrate the frequency histograms which were taken from the experiments for defining input membership functions[3,7].

From these experimental results we can obtain a set of data from the *River group* part (see Figure 8 and 9) to define the membership functions. Triangular membership function is most common and has proved to be a good compromise between effectiveness and efficiency. Overlapping between fuzzy-set boundaries is desirable and the key to smooth operation of the system. To simplify the procedure of defining fuzzy membership functions, an overlap of 50 percent between adjacent fuzzy sets is used in this experiment. In addition to each fuzzy set the central value and the slopes on either side are chosen. Figure 10 shows the fuzzy sets associated with the inputs and output of the system.

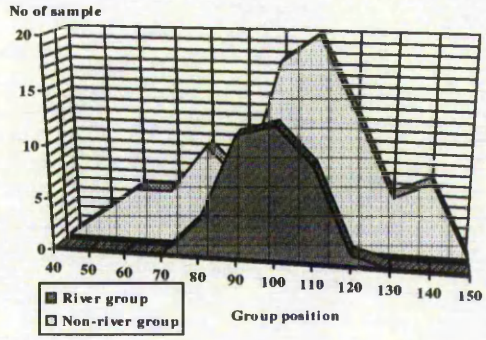


Figure 8: *Frequency histogram for the positions of pixel groups*

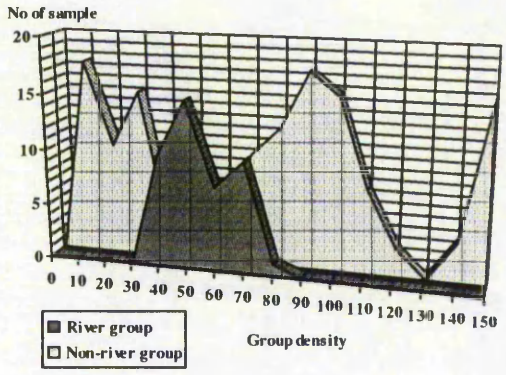


Figure 9: *Frequency histogram for the densities of pixel groups*

2.2 Fuzzification process

Fuzzification is the procedure of calculating an input value to represent a degree of membership in one or more fuzzy sets. This process uses two basic steps which are repeated for each system input. First, a crisp input has to be read and scaled to a value between 0 and 100. Second, the input must be translated to a degree of membership function. Figures 11 and 12 show two system inputs, *position* and *density*. Each value of system input has a degree of membership in each of these sets. Once the degrees of memberships are assigned, we can utilise these values to evaluate the rules.

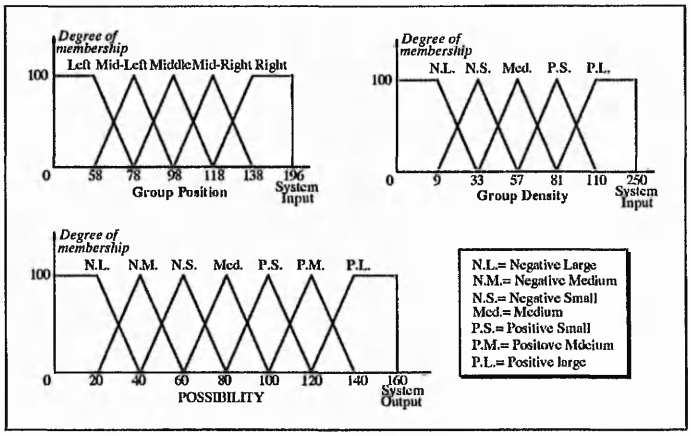


Figure 10: *System input and output membership functions*

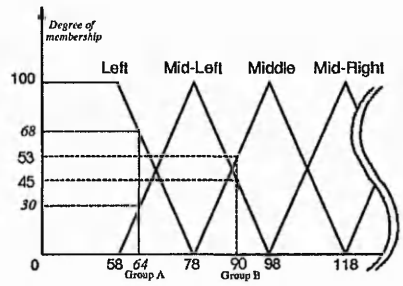


Figure 11: *Fuzzy sets for "group position"*

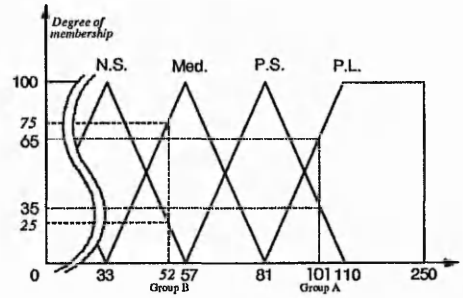


Figure 12: *Fuzzy sets for "group density"*

2.3 Inference and composition

Fuzzified inputs are processed through a pre-defined set of rules using min-max evaluation to form fuzzified outputs. The author developed a set of rules that have the form of

IF [*antecedent_one*] AND [*antecedent_two*]
THEN [*consequence*]

which are listed in Figure 13. The antecedents of rules correspond directly to degrees of membership calculated during the fuzzification process. Each antecedent has a degree of truth assigned to it as a result of fuzzification.

Rule 1:	IF position is Left	AND	density is N.L.	THEN	possibility is N.M.
Rule 2:	IF position is Left	AND	density is N.S.	THEN	possibility is P.S.
Rule 3:	IF position is Left	AND	density is Med.	THEN	possibility is P.S.
Rule 4:	IF position is Left	AND	density is P.S.	THEN	possibility is N.M.
Rule 5:	IF position is Left	AND	density is P.L.	THEN	possibility is N.L.
Rule 6:	IF position is Mid-Left	AND	density is N.L.	THEN	possibility is P.S.
Rule 7:	IF position is Mid-Left	AND	density is N.S.	THEN	possibility is P.M.
Rule 8:	IF position is Mid-Left	AND	density is Med.	THEN	possibility is P.L.
Rule 9:	IF position is Mid-Left	AND	density is P.S.	THEN	possibility is Med.
Rule 10:	IF position is Mid-Left	AND	density is P.L.	THEN	possibility is N.L.
Rule 11:	IF position is Middle	AND	density is N.L.	THEN	possibility is P.S.
Rule 12:	IF position is Middle	AND	density is N.S.	THEN	possibility is P.M.
Rule 13:	IF position is Middle	AND	density is Med.	THEN	possibility is P.L.
Rule 14:	IF position is Middle	AND	density is P.S.	THEN	possibility is P.S.
Rule 15:	IF position is Middle	AND	density is P.L.	THEN	possibility is N.L.
Rule 16:	IF position is Mid-Right	AND	density is N.L.	THEN	possibility is P.S.
Rule 17:	IF position is Mid-Right	AND	density is N.S.	THEN	possibility is P.M.
Rule 18:	IF position is Mid-Right	AND	density is Med.	THEN	possibility is P.L.
Rule 19:	IF position is Mid-Right	AND	density is P.S.	THEN	possibility is Med.
Rule 20:	IF position is Mid-Right	AND	density is P.L.	THEN	possibility is N.L.
Rule 21:	IF position is Right	AND	density is N.L.	THEN	possibility is N.M.
Rule 22:	IF position is Right	AND	density is N.S.	THEN	possibility is P.S.
Rule 23:	IF position is Right	AND	density is Med.	THEN	possibility is P.S.
Rule 24:	IF position is Right	AND	density is P.S.	THEN	possibility is N.M.
Rule 25:	IF position is Right	AND	density is P.L.	THEN	possibility is N.L.

Figure 13: *System rule base*

In inference and composition processes, strengths are enumerated based on antecedent values and then assigned to the rules' output strengths. Figure 14 illustrates the actual fuzzy outputs calculated during the rule evaluation process for pixel group A. The strength of a rule is assigned the value of the weakest (*minimum*) antecedent. As more than one rule applies to the same specific action, the strongest (*maximum*) value of rules is used :

i) from Rule 4:

$N.M._{rule\ strength}$
$= min\ (antecedent\_one, antecedent\_two)$
$= min\ (68, 35) = \underline{35}$

ii) Rule 5:

$N.L._{rule\ strength1} = \min (68, 65) = 65,$

from Rule 10 also

$N.L._{rule\ strength2} = \min (30, 65) = 30,$

the maximum rule strength on fuzzy set N.L. is

$N.L._{rule\ strength} = max\ (65, 30) = \underline{65}$

iii) Rule 9:

$Med._{rule\ strength} = min\ (30, 35) = \underline{30}$

In order to further improve the speed of this calculation, the Fuzzy Associative Memory (FAM) Bank[1] is applied to reduce the number of the rules. Inspecting the FAM Bank (Figure 15), the following fuzzy system rule can be formulated:

from rule (A) indicated in Figure 15,

IF the *Group Position* is *Right*
AND the *Group Density* is *Positive Small*
THEN the *Possibility* is *Negative Medium*

This FAM Bank is comprised of 5 × 5 rules. We can reduce the 25 rules per FAM Bank to 11 rules per table by compounding the rules in the Bank. For instance, rule (b) indicated in Figure 15 merges three [*antecedent one*]s of the rules to take the form:

from rule (B),

IF the *Group Position* is *Near Middle*
AND the *Group Density* is *Negative Small*
THEN the *Possibility* is *Positive Medium*

2.4 Defuzzification process

The defuzzification process is to convert its fuzzy outputs into a single raw or crisp output. There are more than 30 valid defuzzification methods. In these experiments, we choose the *"centre-of-gravity method"* which is a common and accurate defuzzification technique for resolving both the vagueness and conflict issues[8]. Figure 16 is used to illustrate the defuzzification of the output using the centre of gravity method:

Rule1:	IF position is 68	AND	density is 0	THEN possibility is N.M.
Rule2:	IF position is 68	AND	density is 0	THEN possibility is P.S.
Rule3:	IF position is 68	AND	density is 0	THEN possibility is P.S.
Rule4:	IF position is 68	AND	density is 35	THEN possibility is N.M.
Rule5:	IF position is 68	AND	density is 65	THEN possibility is N.L.
Rule6:	IF position is 30	AND	density is 0	THEN possibility is P.S.
Rule7:	IF position is 30	AND	density is 0	THEN possibility is P.M.
Rule8:	IF position is 30	AND	density is 0	THEN possibility is P.L.
Rule9:	IF position is 30	AND	density is 35	THEN possibility is Med.
Rule10:	IF position is 30	AND	density is 65	THEN possibility is N.L.
Rule11:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule12:	IF position is 0	AND	density is 0	THEN possibility is P.M.
Rule13:	IF position is 0	AND	density is 0	THEN possibility is P.L.
Rule14:	IF position is 0	AND	density is 35	THEN possibility is P.S.
Rule15:	IF position is 0	AND	density is 65	THEN possibility is N.L.
Rule16:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule17:	IF position is 0	AND	density is 0	THEN possibility is P.M.
Rule18:	IF position is 0	AND	density is 0	THEN possibility is P.L.
Rule19:	IF position is 0	AND	density is 35	THEN possibility is Med.
Rule20:	IF position is 0	AND	density is 65	THEN possibility is N.L.
Rule21:	IF position is 0	AND	density is 0	THEN possibility is N.M.
Rule22:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule23:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule24:	IF position is 0	AND	density is 35	THEN possibility is N.M.
Rule25:	IF position is 0	AND	density is 65	THEN possibility is N.L.

Figure 14: *Inference and composition for pixel group A*

Rule (A): IF Position is Right and Density is P.S.
THEN Possibility is Negative Medium
Rule (B): IF Position is Near Mid. and Density is N.S.
THEN Possibility is Positive Medium

Poss-ibility		Position				
		Left	ML	Mid	MR	Right
Density	NL	NM	PS	PS	PS	NM
	NS	PS	PM (E)	PM	PM	PS
	Med	PS	PL	PL	PL	PS
	PS	NM	Med	PS	Med	NM (A)
	PL	NL	NL	NL	NL	NL

Figure 15: *Fuzzy Associative Memory (FAM) Bank to determine the possibility*

i) A centroid point on the x axis is found for each output membership function;
ii) The membership functions are limited in height by the applied rule strength;
iii) The areas of the membership functions are calculated;
iv) The defuzzified outputs are derived by weighted averages of the centroid points and the enumerated areas:

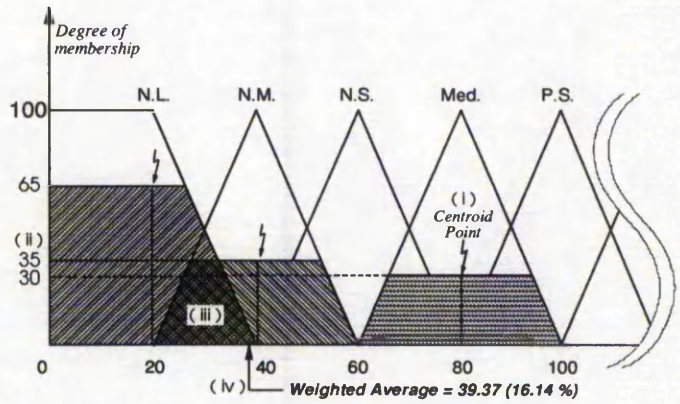


Figure 16: *Defuzzification process for pixel group A*

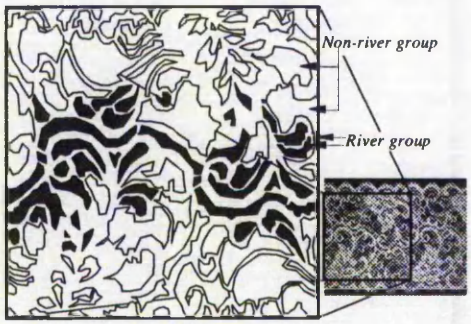


Figure 17: *Each possible river segments whose weights are bigger than 80 (50%)*

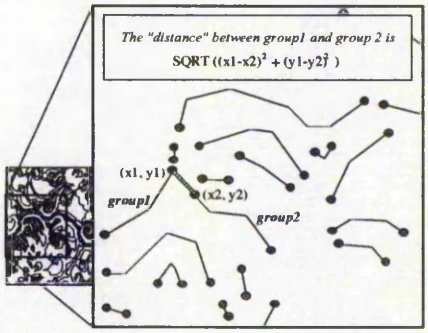


Figure 18 : *An example for calculating the distance between pixel groups*

$$Weighted\ average = \frac{\sum(shaded\ area \times centroid\ point)}{\sum(shaded\ area)}$$

By relying on the use of fuzzy inference technique, each black pixel group could be calculated and assigned an average weight (*possibility*). For instance, in Figure 7, the output value for group A is 39.37 (*24.61 %*) (refer to Figure 16), also the group B is 134.64 (*84.15 %*). Since the average weight of group A is only 16.14% (less than 50%), this means that the pixel group only has a 16 percent *possibility* of being a segment of the river. It is, therefore, concluded that group A is not a part of the river.

2.5 Verification

Once *all* the black pixel groups have been assigned a *possibility value* (average weight), the pixel groups whose possibility values are less than 80 (50%) are abandoned (see Figure 17). The verification process can then be broken down into the following tasks:

i) Calculate the distance between two adjacent groups;
ii) If the distance is shorter than a specified value (set to six pixels long in these experiments) a network is built to record this path;
iii) Continuously trace the distances between pixel groups while recording all the correct paths until a new pixel group reaches the border of the image (right hand edge of the frame);
iv) Calculate the total possibility values and divide by the number of the group in this path (*average possibility*);

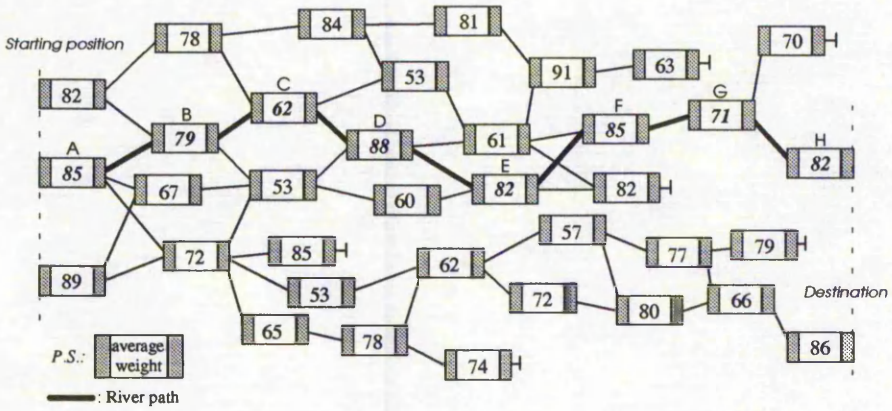


Figure 19: *Interconnection between each possible river segments*

v) If the *average possibility* is bigger than a specified value, (110 or 75% was used in the experiments) then the correct river has been found; if the average possibility is less than this value, repeat step (iii) to (v) until the correct river is located.

Figure 18 illustrates the computation of the distance between two adjacent pixel groups. By calculating the distances and tracing the average possibilities in all these segments, the river location, highlighted in Figure 19, can be pin-pointed.

3. EXPERIMENTAL RESULTS

A number of experiments were carried out to evaluate the effectiveness of this method. Various kinds of lace patterns were employed for detecting the river location. All cutting paths across the patterns were successfully found. The time taken to isolate the river and produce cutting path depends on complexity of the pattern. Time taken for most kinds of motif is typically about 300 milli-seconds using an Intel 80486 processor running at 66 MHz. However, in the case of a very few intricate lace patterns (e.g. Figure 20), up to 1.5 seconds is required. Once the river path on the first frame is found, this knowledge can be utilised to speed up the detection for the river in subsequent frames to meet the real-time requirements of the system. Some sample lace patterns together with the resulting river path are shown in Figures 20 to 22.

Figure 20: *Example A of river extraction*

The strip of lace is likely to stretch or contract while it is passed through the machine via the feed mechanism. Several experiments have been carried out for investigating the capability of this approach. Various kinds of lace patterns have

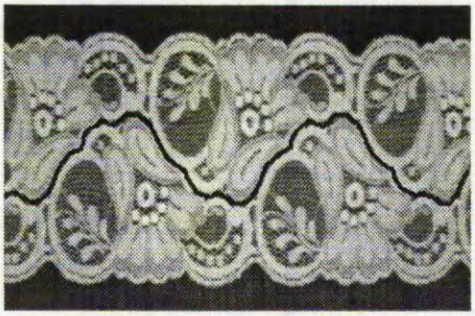


Figure 21: *Example C of river extraction*

Figure 22: *Example B of river extraction*

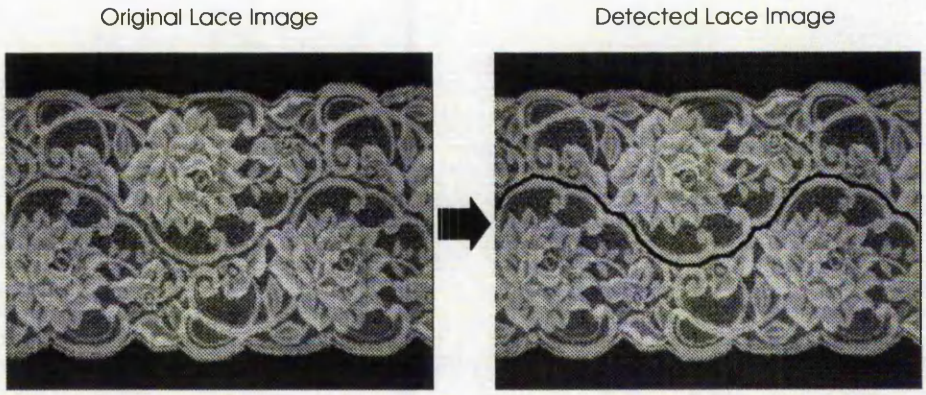


Figure 27: *Lace pattern under maximum stretch*

5. FUTURE WORK

Remote sensing will be used to post-process the scalloped lace in real-time in order to monitor the effectiveness of path finding and cutting process downstream of the cutter. A tight coupling between the two allows detection of errors and alteration of control parameters on line.

6. ACKNOWLEDGEMENT

This work has been carried out in collaboration with PACER Systems Ltd., unit 6, Robin Hood Industrial Estate, Nottingham, NG3 1GE, United Kingdom.

7. REFERENCES

1. Nedungadi, Ashok and Wenzel, Dennis J. "A novel approach robot control using fuzzy logic", IEEE, ISSN#0-7803-0233-8/91, 1991.
2. Norton-Wayne, L. "Inspection of lace using machine vision", Computer Graphics Forum, vol. 10, iss.2, June 1991, pp 113 - 19.
3. Roberts, D.W. "Analysis of forest succession with fuzzy graph theory", Ecological Modeling, 45:261-274, 1989.
4. Russell, R.A. and Wong, P. "Automation of lace cutting using computer vision", Robots: Coming of Age. Proceedings of International Symposium and Exposition on Robots. Designed the 19th ISIR by the International Federation of Robotics, Nov. 1988, pp 385-93.
5. Sherkat, Nasser; Birch, Mike and Thomas, Peter. "Real-time vision for automatic lace cutting", IS&T/SPIE Symposium on Electronic Imaging Science & Technology, 6-10 February 1994.
6. Sherkat, Nasser; Chi-Hsien V. Shih and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Application, kawasaki, Japan, Dec. 13-15, 1994. (to be published)
7. Turksen, I.B. "Measurement of fuzziness: interpretation of the axioms of measure", Proceeding of the Conference on Fuzzy Information and Knowledge Representation for Decision Analysis, IFAC, Oxford, 1984, pp 97-102.
8. Zadeh, Lofti A. "Fuzzy logic", Computer, IEEE, April 1988, pp 83 - 93.
9. Zimmermann, H. J. Fuzzy Sets, Decision Making, and Expert Systems, Kluwer Academic Publishers, Lancaster, 1987.

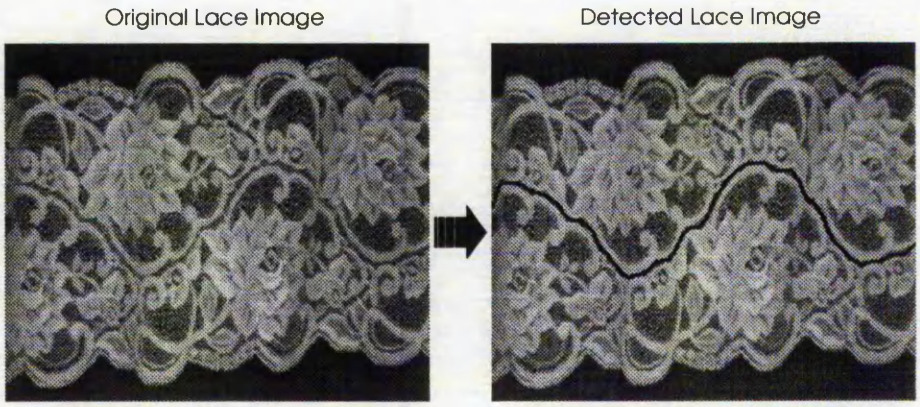


Figure 25: *Lace pattern under 40% contraction (successfully detected)*

Lace was stretched lengthwise in order to emulate stretch resulting from lace transport. Figure 27 shows that the lace pattern has been stretched as much as possible, and its detected river across the pattern. Typically, about 15 to 30 per-cent of lace can be stretched, depending on material and patterns.

Compared with the previously reported method[5], the fuzzy logic based approach is more effective. The traditional image processing technique for finding the river heavily depends on the repeat of the lace pattern. In other words, the two extremes of the river should be equi-distant from their nearest edge, and after a distance equal to the *repeat period* of the design, the river should be back at the same position relative to the two edges as it was at the start. When lace is distorted, these features of the river are absent. That is why the conventional method fails when strips of lace are slightly distorted (5- 10%).

4. CONCLUSION

In the preceding sections of this paper, we have described the development of a fuzzy reasoning rule-based system for detecting various kinds of lace patterns. Experimental results indicate that the objectives have mostly been fulfilled. The system requires no prior knowledge of any particular lace pattern or any training. According to the results of the experiments this method can successfully detect the river path within varied lace patterns. Compared with the conventional image processing methods,[4,5] it is not only easier to design and implement the system by means of using the fuzzy reasoning techniques, but also more effective in coping with distortion.

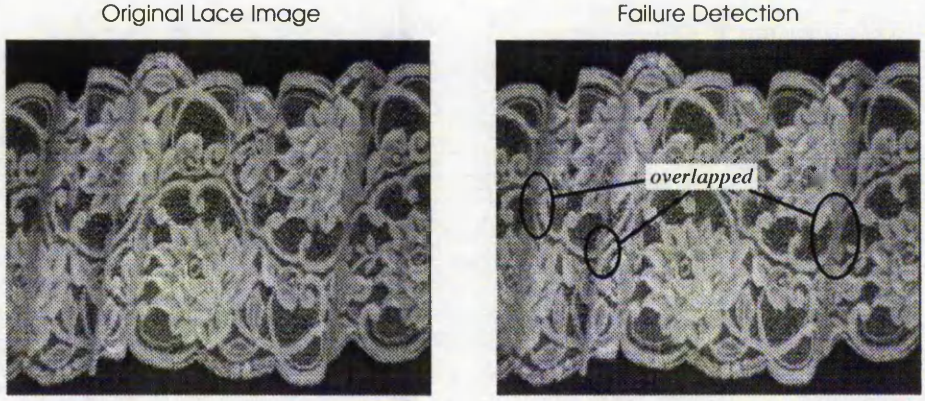


Figure 26: *Lace pattern under more than 50% contraction (fail to detect the correct cutting path)*

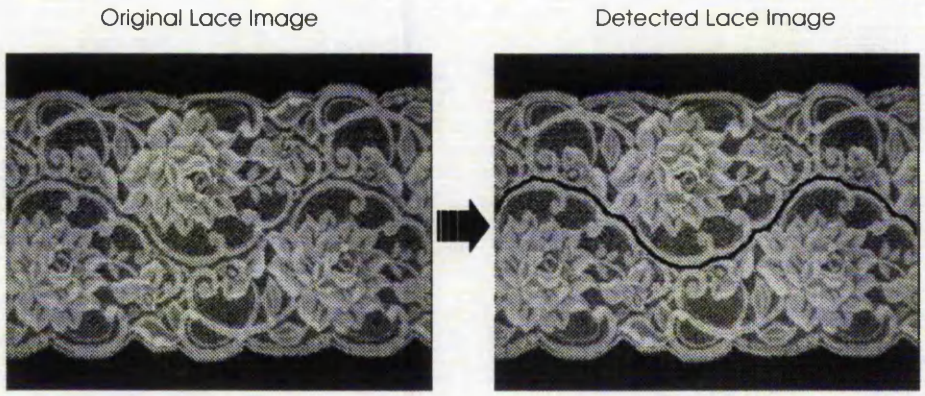


Figure 27: *Lace pattern under maximum stretch*

5. FUTURE WORK

Remote sensing will be used to post-process the scalloped lace in real-time in order to monitor the effectiveness of path finding and cutting process downstream of the cutter. A tight coupling between the two allows detection of errors and alteration of control parameters on line.

6. ACKNOWLEDGEMENT

This work has been carried out in collaboration with PACER Systems Ltd., unit 6, Robin Hood Industrial Estate, Nottingham, NG3 1GE, United Kingdom.

7. REFERENCES

1. Nedungadi, Ashok and Wenzel, Dennis J. "A novel approach robot control using fuzzy logic", IEEE, ISSN#0-7803-0233-8/91, 1991.
2. Norton-Wayne, L. "Inspection of lace using machine vision", Computer Graphics Forum, vol. 10, iss.2, June 1991, pp 113 - 19.
3. Roberts, D.W. "Analysis of forest succession with fuzzy graph theory", Ecological Modeling, 45:261-274, 1989.
4. Russell, R.A. and Wong, P. "Automation of lace cutting using computer vision", Robots: Coming of Age. Proceedings of International Symposium and Exposition on Robots. Designed the 19th ISIR by the International Federation of Robotics, Nov. 1988, pp 385-93.
5. Sherkat, Nasser; Birch, Mike and Thomas, Peter. "Real-time vision for automatic lace cutting", IS&T/SPIE Symposium on Electronic Imaging Science & Technology, 6-10 February 1994.
6. Sherkat, Nasser; Chi-Hsien V. Shih and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Application, kawasaki, Japan, Dec. 13-15, 1994. (to be published)
7. Turksen, I.B. "Measurement of fuzziness: interpretation of the axioms of measure", Proceeding of the Conference on Fuzzy Information and Knowledge Representation for Decision Analysis, IFAC, Oxford, 1984, pp 97-102.
8. Zadeh, Lofti A. "Fuzzy logic", Computer, IEEE, April 1988, pp 83 - 93.
9. Zimmermann, H. J. Fuzzy Sets, Decision Making, and Expert Systems, Kluwer Academic Publishers, Lancaster, 1987.

Tightly Coupled Vision Based Control System Using Inexact Algorithms

Nasser Sherkat[*], *Chi-Hsien V. Shih*[†], *Peter Thomas*[‡]

The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom
Fax.: +44 115 9486518

[*]*Tel.:* +44 115 9418418 ex.2262 [*]*E-mail:* ns@doc.ntu.ac.uk
[†]*E-mail:* vsh@doc.ntu.ac.uk , [‡]*E-mail:* pdt@doc.ntu.ac.uk

Abstract

A novel approach to deal with problems in handling flexible materials has been described.. A number of solutions to this problem have been developed by innovative combination of fuzzy logic and neural networks (inexact algorithms). Using the pre- and post-processing vision systems, it is possible to monitor the lace scalloping process as well as generating on-line information for the Artificial Intelligence engines. This allows overcoming the problems of material distortion due to the trimming operation. A *Spring Mounted Pen (SMP)* is used in experiments to emulate the distortion of lace pattern caused by tactile cutting and feed mechanism misalignment. Applying the algorithms developed, the system can automatically compensate for flexibility and produce excellent outcome better than a human operator.

1 Introduction

A number of attempts have been made to automate the process of lace scalloping and quality inspection [1][2][3][4]. Work has been reported in using laser technology to cut deformable materials [5]. Although using laser reduces this deformation, distortion due to mechanical feed misalignments persists. Changes in the lace pattern are also caused by the release of tension in the lace structure as it is cut. In order to tackle the problem of distortion due to material flexibility in general, a novel approach using *inexact algorithms*, e.g., *fuzzy logic*, *neural networks* and *neural fuzzy technique*, have been developed and described here.

As depicted in Figure 1, a spring is used to mount a pen onto the Z axis of the machine (*Spring Mounted Pen*). This device is employed in the experiments to emulate the movement of the lace strip due to the cutting forces caused by the tactile cutter and feed misalignment. A black line is drawn on paper to emulate the river path within a lace strip. The pre-processing vision system captures an image of the paper strip and stores it in memory. A pattern (desired cutting path) is extracted from the image and transferred to the machine console. This information is used to guide the SMP to draw a second line on the paper strip. As the processed object is passed under the camera of the post-processing vision system, an image is taken and analysed to inspect the "quality" of this drawing. The host system takes this fed-back information and determines suitable correcting actions to improve this operation.

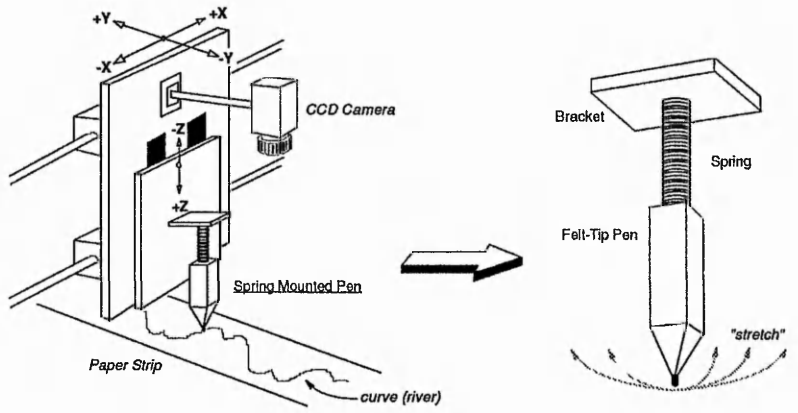


Figure 1: *Spring Mounted Pen (SMP) connected with the testing rig*

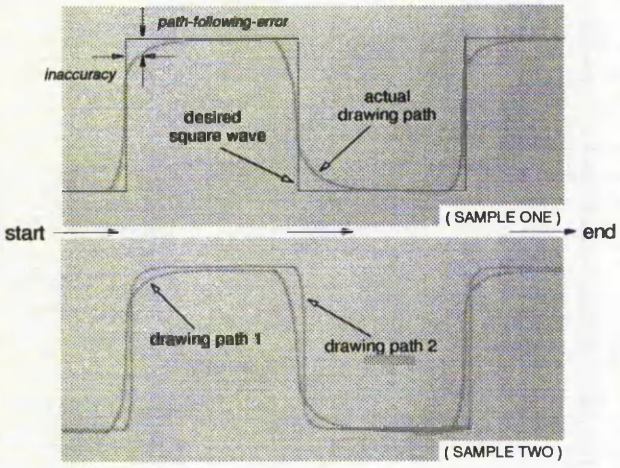


Figure 2: *Samples of square wave following process*

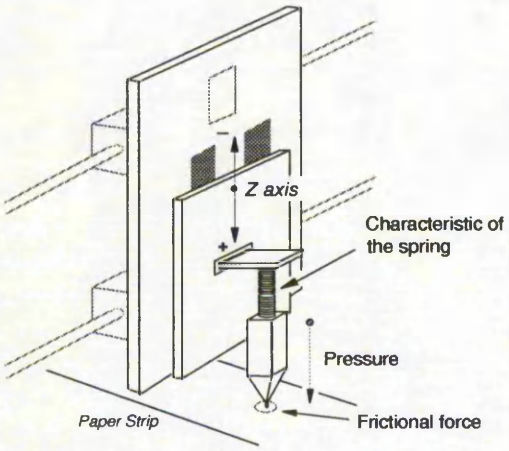


Figure 3: *Three system coefficients affects the path following process*

2 Problem Analysis

Figure 2 represents the results of following a square wave using the SMP. Due to the inherent characteristics of the spring, the *path-following-errors* appear between the desired path and the actual target line. Each time the pen is put in contact with paper, the axial load on the spring changes. This consequently causes the pattern generated by the SMP to alter (*path1* and *path2* indicated in Figure 2).

It is clear to observe that the path-following-error will be generated when the direction of the drawing is changed - the larger the angular variation of the path following, the larger is the error. The amount (magnitude) of the path-following-error generated is depends on the *characteristic of the spring* engaged, the *pressure* on the SMP, and the *frictional force* in between the tip of the pen and the paper (refer to Figure 3). As any one of the system coefficients is altered, the result of the SMP drawing will be entirely different.

As shown in Figure 4, while the drawn path is transported beneath the camera of the post-processing vision system, an image of the paper strip is captured and stored in the memory. The path-following-error appearing in this frame is detected. Since the extreme complexity of the path-following-error caused by the SMP is affected by the three system parameters mentioned above, it seems to be very difficult to use conventional methods, where physical sensors are applied to measure all of the system parameters hence find the related information to correct this error. In order to overcome the problems of complexity, the inexact algorithms are considered and introduced here.

3 Proposed Method

An innovative method based on modeling human

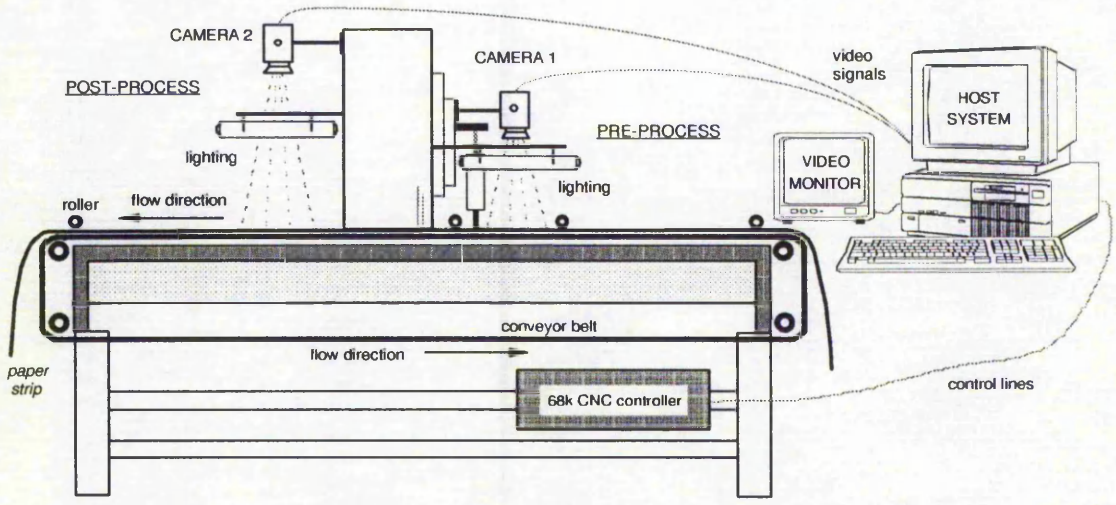


Figure 4: *Schematic diagram of the testing rig*

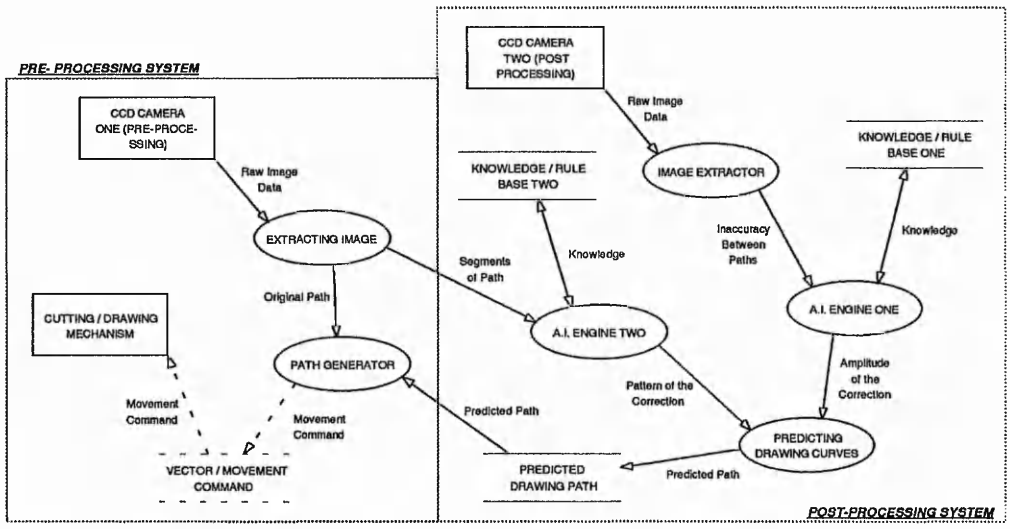


Figure 5: *The overview of the vision and motion control systems*

operators' experience and control actions using fuzzy logic and neural networks are developed to solve the problem. By translating a skilled operator's knowledge into a set of linguistic terms or groups of network connections, the intelligent machine console can learn from the experiences (on-line learning) and self-adjust the control actions to match the desired objective. Figure 5 shows a data flow diagram for the system overview. Two A.I. engines are constructed in the system in order to determine a suitable correction added to the original path. This correction is separated into two "energies":

1) Correcting pattern; and
2) Correcting amplitude.

The path-following-error is detected and fed into *A.I. Engine One* which analyses the difference between the paths also decides the *amplitude of correction* for further processing. Camera One, as indicated in Figure 5, is triggered to capture a new frame of the desired path on paper. At this time, before the extracted curve is sent to the *path generator*, the segments of the extracted path are passed to *A.I. Engine Two* which determines the correcting *pattern*. Both the *amplitude* and the *pattern* are utilised to generate a *predicted correcting (compensated) path*. Finally, the path generator reads the predicted path and the original path to produce the machine movement data.

According to the information provided by the A.I. Engines, the machine guides the SMP to follow the desired path. A small path-following-error may still appear in the first corrected frame. Using the scheme stated above the error information is repeatedly supplied to the A.I. Engines to calculate more accurate correcting actions until two paths are finally matched (learning process). Two schemes based on examining each individual segment of the path have been designed and

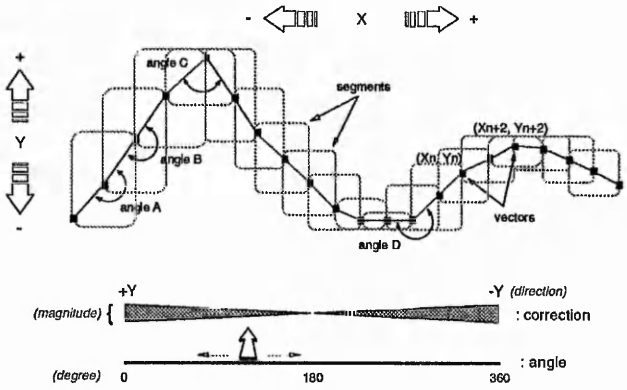


Figure 6: *Correlation between angle and the correcting energy (direction and magnitude)*

implemented. The first method is based on manipulating the angles between *three* consecutive *vectors* in the path *(3VMethod)*. The second method is based on the analysis of the angles between the *two* adjacent *vectors* and the X/Y coordinates *(2VMethod)*.

3.1 3VMethod

Figure 6 illustrates the correction between the angle and the correcting magnitude. As the angle between these three consecutive vectors is small, the correction added on the desired path is large. Yet if the angle is large (near 180 degrees), the correction is small. The following Equations represent this concept.

$$CompensatedPath = \sum_{i=1}^{No\ of\ Segments} Segment(i)_{corrected}$$

$$Segment(i)_{corrected} = Segment(i)_{original} + \sum_{n=1}^{i} (correcting\ energy)_n$$

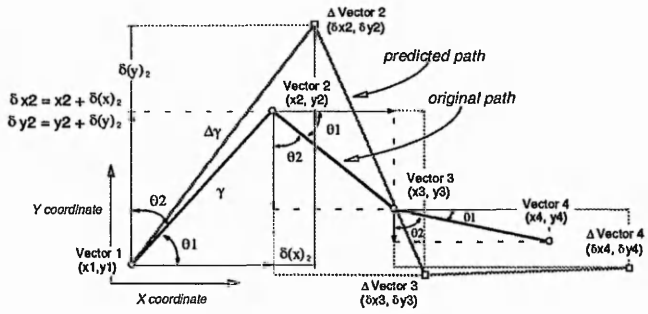


Figure 7: *Calculating a new corrected vector using 2VMethod*

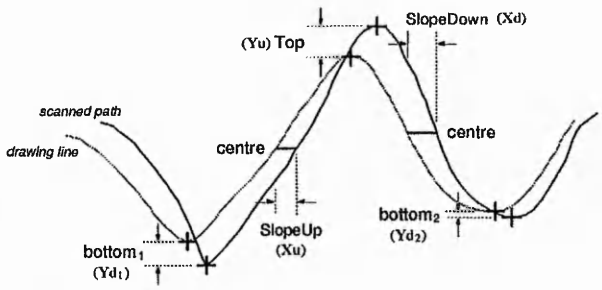


Figure 8: *Detecting the inaccuracy of path following*

3.2 2VMethod

Applying this approach, every *two* consecutive vectors are analysed over the entire path. As a straight line is connected from *Vector 1* to *Vector 2* (refer to Figure 7), the angles (θ1 and θ2) between the line (γ) and the X / Y coordinates are used to compute the possible correcting energies *(δ(x) and δ(y))*. Figure 7 represents the calculation. The angles θ1 and θ2 are complementary. These two angles are passed to an A.I. engine which is designed by means of using inexact methods to determine the correcting energies (δ(x) and δ(y)). The prediction of the new estimated vector *(ΔVector2(δx2, δy2))* is calculated by Equation 3.2-1. Equation 3.2-2 describes the procedure of computing the pattern of the predicted path.

$$\delta x(i) = x(i) + \delta(x)_i, \quad \delta y(i) = y(i) + \delta(y)_i \qquad (3.2\text{-}1)$$

$$\begin{aligned} Predicted\,Path &= Vector\,1 + \sum_{i=2}^{n} \Delta Vector(i) \\ &= \{x1, y1\} + \sum_{i=2}^{n} \{\delta x(i), \delta y(i)\} \end{aligned} \qquad (3.2\text{-}2)$$

where i is the i^{th} segments in a path and n is the number of vectors in the path.

Once the pattern of correction is determined, the next step is to decide the amount of the amplitude needed for the correction. As the first processed frame is passed under the post-processing vision system, an image is taken and sent to the host system. A software recogniser is employed to distinguish two different colour lines on paper. The top, bottom and centre positions in both paths are taken to measure the inaccuracy of the SMP following. The distances between these points within the different paths (Figure 8) are calculated and passed to an A.I. Engine. The engine takes the data and calculates the average errors for each of the parameters - *Top, Bottom, SlopeUp,* and *SlopeDown*. These parameters are then used by the A.I Engine One to determine a suitable amount of amplitude for the correction.

Three different techniques based on the inexact algorithms, such as fuzzy logic, neural fuzzy theory, and neural networks, have been applied to determine the correcting energies (pattern and amplitude). The detailed description of applying the inexact algorithms to create the correcting energies can be found in [6].

4 Experiment Results

A working prototype is constructed. A number of experiments have been carried out to evaluate the effectiveness of this algorithm. This approach has been implemented using fuzzy logic, neural networks and neural fuzzy technique respectively.

Figure 9 shows an example of applying 3VMethod in the SMP following process. Compared to the pattern drawn without the correction, about 60 to 80 percent of the path-following-error has been successfully removed by applying the 3VMethod. The results of SMP following using 2VMethod is illustrated in Figure 10. Through the self-adapting process, the intelligent console can automatically make the appropriate compensation. According to various experiments, after three frames of correction almost all the errors caused by the spring can be eliminated. Figure 11 depicts two different correcting paths created by the neural fuzzy engine during the self-learning process.

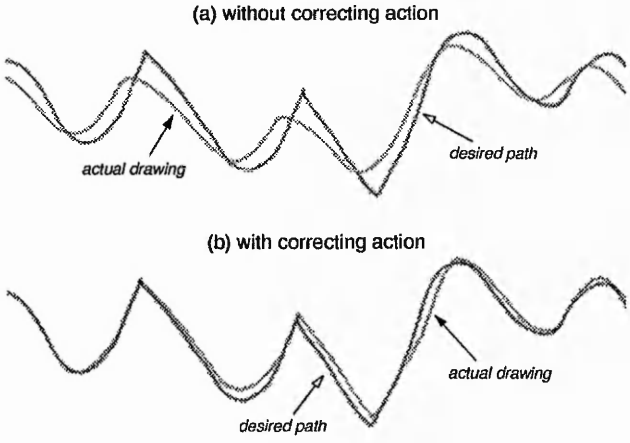


Figure 9: *Example of using 3VMethod to correct the error*

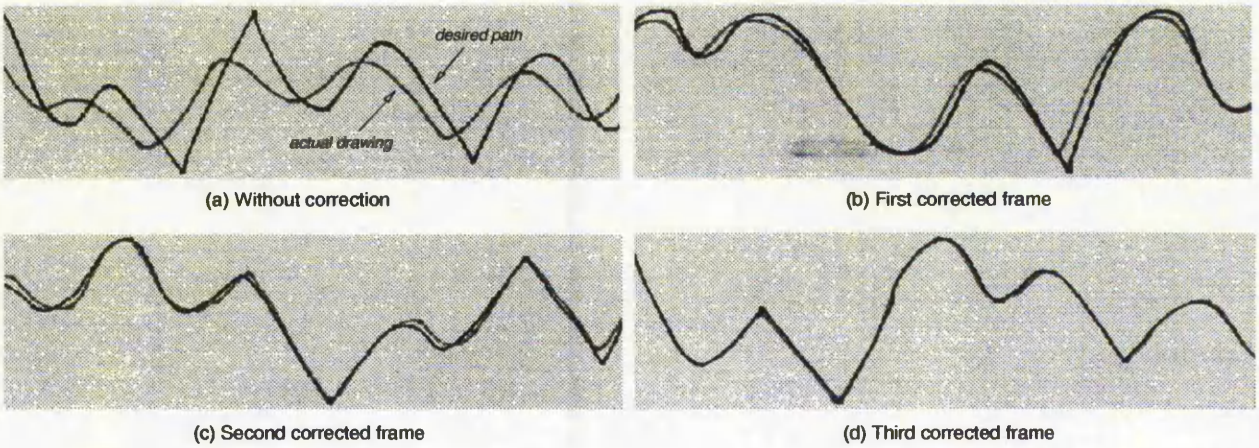


Figure 10: *Samples of using 2VMethod to correct the error*

5 Conclusion

In the preceding sections of this paper, we have described attempts to develop an intelligent tightly coupled vision based control system using inexact algorithms. The development of the system is a novel approach to material processing and has further applications where modeling system behaviour characteristics is difficult, such as controlling a robot moving on a slippery surface, drive a car on snow, or piloting a boat, etc. Only less than three frames of learning process are required before the machine reacts correctly in minimising the error. According to various experimental results, the developed system can deal with any irregular shape of path, and can produce excellent outcome better than a human operator.

6 Acknowledgments

This work has been carried out in collaboration with Pacer Systems Ltd. and Axiomatic Technology Ltd.

7 References

[1] Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology. California, USA, 1995.

[2] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE Fifth International Conference on Image Processing and Its Applications. Edinburgh, U.K., 1995.

[3] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Automation of lace cutting using real-time vision", 8th International Congress on Condition Monitoring and Diagnostic Engineering Management. Canada, 1995.

[4] Norton-Wayne. L. "Inspection of lace using machine vision", Computer Graphics Forum. Vol.10, 1991.

[5] King, T. "Real-time tracking of patterns on deformable materials using DSP", IEE International workshop on systems engineering for real-time applications. Royal Agricultural College, Cirencester, U.K, 1993.

[6] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Close coupling of pre and post processing vision stations using inexact algorithms", IS&T/SPIE Symposium on Electronic Imaging Science & Technology. USA, 1996.

Amplitudes of Correction

	Yu	Yd	Xu	Xd
Desired Path	0	0	0	0
Drawing Path 1	-420	260	-100	-140
Drawing Path 2	-970	800	-450	-480

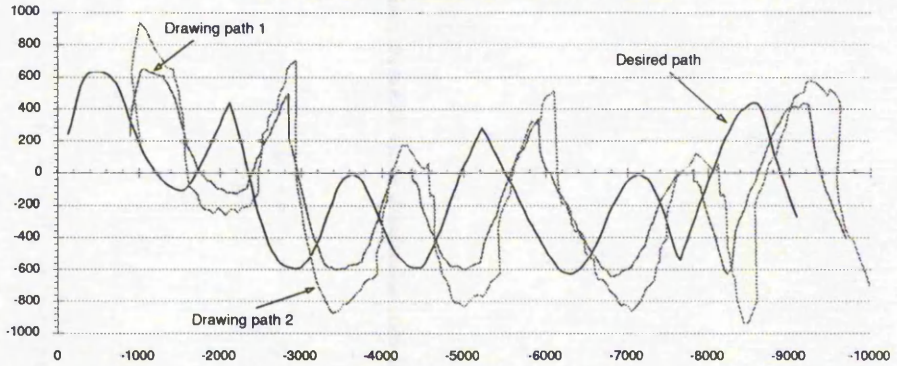


Figure 11: *Two correcting paths created by the neural fuzzy engine using 2VMethod*

A VISION BASED CONTROL SYSTEM USING NEURAL FUZZY THEORY

Nasser Sherkat, Chi-Hsien V. Shih, Peter Thomas
The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom

ABSTRACT

During the last few years automating lace handling has received much attention in the United Kingdom. Research and development is ongoing in the two main areas of automatic lace cutting and automatic lace quality inspection. This paper reports on research and development work under gone at the Nottingham Trent University, in collaboration with local industry, in automation of lace scalloping. The research is reviewed in its various stages of development. An innovative scheme, based on neural fuzzy theory, for dealing with the problem of lace distortion, due to its flexible nature, is described.

Keywords: machine vision, fuzzy logic, neural networks, neural fuzzy theory, spring mounted pen, on-line self-learning.

1. INTRODUCTION

Lace, intended for decoration of clothing and furniture, is mass produced in wide rolls. In order to increase the production rate, the rolls are made by repeating the desired lace strips side-by-side (Figure 1). At the second manufacturing stage the repeated strips are separated along pre-designated paths (rivers) which run throughout the length of the roll. The process of trimming the lace has been carried out manually. This is a lengthy and expensive process and results in slowing the rate of production. A small number of machines have been developed that use a simple passive cutter mechanism that relies on the structural strength of the lace pattern. In such systems the cutter is held stationary while the lace is run against it. This approach is only suitable in cases where the lace pattern is of very shallow scallop. In the case of deep scallop patterns a more sophisticated method of guidance, based on an

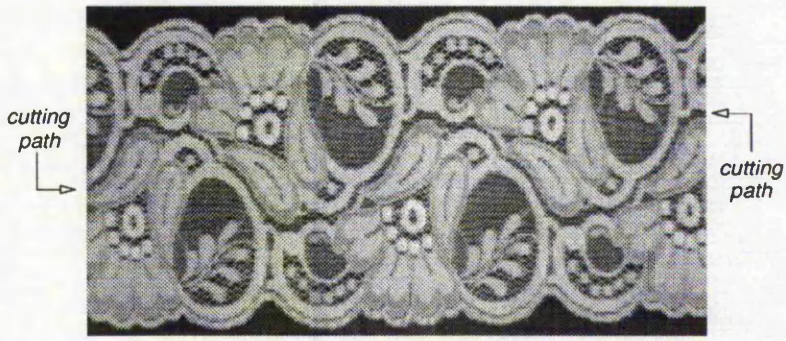


Figure 1: *A typical lace pattern*

active cutter, is required.

A vision based system has been developed for industrial lace scalloping. On-line pattern recognition is performed to detect the cutting path, which is vectorised and transferred to a trimming mechanism. In order to satisfy industrial requirements two main conditions must be satisfied. To achieve a sufficient degree of automation, first, the river must be found without prior knowledge of the lace pattern scanned. A *fuzzy pattern recognition technique* has been developed to detect varied shapes of rivers within the lace patterns. Secondly, finding of the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach called the *Line Mapping Method* is used to speed up the search for the river in subsequent frames.

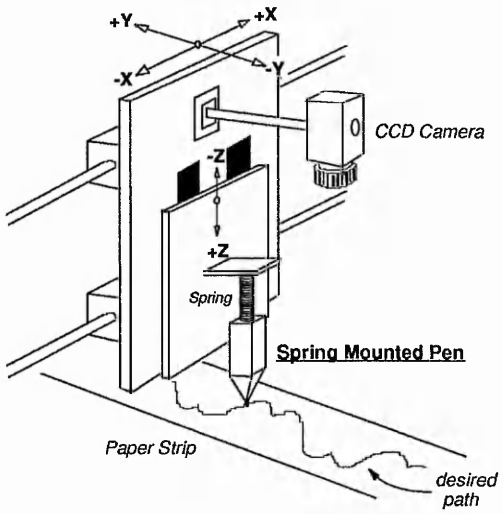


Figure 2: *Spring Mounted Pen connected with the test rig*

Work has been reported in using lasers to cut deformable materials [1]. Although using lasers reduces this deformation, distortion due to mechanical feed misalignments persists. In addition using laser is not free from problems and a tight control is of laser parameters is required in order to achieve a satisfactory level of performance. Changes in the lace pattern are caused also by release of tension in the lace structure as it is cut. Our experiments with lace handling have pointed out that the biggest problem to overcome is that of non-linear behaviour due to flexibility. To tackle the problem of distortion due to material flexibility in general, a novel approach, using *fuzzy logic*, *neural networks* and *neural fuzzy theory*, has been developed. A *Spring Mounted Pen (SMP)*, Figure 2, is used in the experiments to emulate the distortion of lace pattern caused by tactile cutting. Using the machine vision station and the intelligent software kernels, it is possible to monitor the scalloping process as well as generating on-line information fed back to the host system. This allows overcoming the problems of lace distortion due to the trimming operation. Two A.I. engines are constructed in the system to determine a compensation fed-back to the controller for correcting the errors. This paper is structured into two main sections: lace pattern detection, and correction of cutting operation.

2. LACE PATTERN DETECTION

2.1 Fuzzy pattern recognition

As a lace pattern is captured by a CCD camera, after a thresholding operation a river shows up as a dark area (pixel group) within the edges that cross from one side of the image to the other in a nearly unbroken sequence. There are *thick threads* that cross the river at intervals. These are indistinguishable from the material surrounding the river (marked by circles in Figure 3). Allowance must be made for small breaks in continuity of the river due to these cross threads. The objective here is to find the river along a lace pattern with no previous knowledge, by using linguistic variables to represent the common feature of the river shape in various lace patterns. These common features may be described as:

a) the *position* of the river is around the *centre* of the pattern;
b) the *density* of the river pixel group is not *high*.

The method for applying fuzzy inference technique to find the cutting path can be divided into the following functional blocks:

- Defining fuzzy membership functions;
- Fuzzification and rule evaluation;
- Defuzzification and verification.

Figure 3: *Bi-level lace bitmap image*

1) Defining fuzzy membership functions

From the linguistic descriptions, two system inputs, *group position* and *group density*, can be defined. By monitoring the position and density of the black pixel groups across a lace pattern, a fuzzy inference engine can determine whether the pixel group is a possible segment of a river. Two initial experiments were carried out to define the fuzzy input and output membership functions. Frequency histograms were used on the sample data to define input membership functions [2]. In this way a set of data can be obtained from the *River group* part to build the membership functions.

Rule (A): IF Position is Right and Density is P.S.
THEN Possibility is Negative Medium
Rule (B): IF Position is Near Mid. and Density is N.S.
THEN Possibility is Positive Medium

Poss-ibility		Position				
		Left	ML	Mid	MR	Right
Density	NL	NM	PS	PS	PS	NM
	NS	PS	PM (B)	PM	PM	PS
	Med	PS	PL	PL	PL	PS
	PS	NM	Med	PS	Med	NM (A)
	PL	NL	NL	NL	NL	NL

Figure 4: *Fuzzy Associative Memory Bank to determine the possibility*

2) Fuzzification and rule evaluation

The fuzzification process computes an input value to represent a degree of membership in one or more fuzzy variables. These fuzzified inputs are processed through a pre-defined set of rules using min-max evaluation to form fuzzified outputs. As depicted in Figure 4, the Fuzzy Associative Memory Bank is applied here to reduce the number of the rules for speeding up the calculation.

3) Defuzzification and verification

Defuzzification process is to convert its fuzzy outputs into a single raw or crisp output. In these experiments, we choose the "centre-of-gravity method" for resolving both the vagueness and conflict issues. The weighted average is calculated as follows:

$$Weighted\ average = \frac{\sum(shaded\ area \times centroid\ point)}{\sum(shaded\ area)} \quad (1)$$

By relying on the use of fuzzy inference technique, each black pixel group could be calculated and assigned an average weight (*possibility*). Once *all* the black pixel groups

have been assigned a *possibility value*, the pixel groups whose possibility values are less than 50% are abandoned. The verification process can then be broken down into the following tasks:

a) Calculate the distance between two adjacent groups;
b) If the distance is shorter than a specified value (set to six pixels long in these experiments) a network is built to record this path;
c) Continuously trace the distances between pixel groups while recording all the correct paths until a new pixel group reaches the border of the image (right hand edge of the frame);
d) Calculate the total possibility values and divide by the number of the group in this path (*average possibility*);
e) If the *average possibility* is bigger than a specified value, (75% was used in the experiments) then the river has been found; if the average possibility is less than this value, repeat step (c) to (e) until the correct river is located.

By calculating the distances and tracing the average possibilities in all these segments, the river, highlighted in Figure 5, can be located.

2.1 Line mapping process

When the first cutting river in the lace strip is successfully detected, the extracted knowledge can be used to speed up the search in subsequent frames. In order to meet the real-time requirements of the system, instead of using traditional pattern matching techniques, a new approach called the *Line Mapping Method (LMM)* has been developed to achieve fast response and higher reliability. This approach is divided into the following processes:

1) Indicating and registering one repeat cutting cycle

A *centre line* is located by calculating the distance between the *upper* and *lower boundaries* shown in Figure 6. Three crossing points between the cutting river and the centre line are marked. The cutting path (river) between the intersections ① and ③ labeled in Figure 6 indicates a repeat cutting cycle, which acts as a *reference path* for detection of subsequent frames.

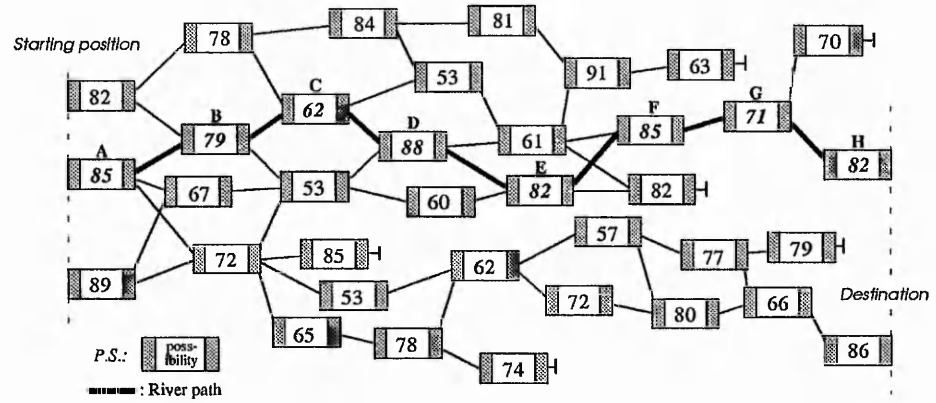


Figure 5: *Interconnection between each possible river segments*

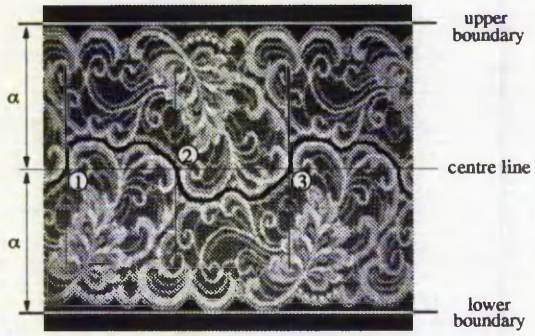


Fig. 6: *Extracting a repeat cutting cycle*

2) Capturing the following frame

The next frame of a 256 grey scale lace image is captured by the frame grabber from the CCD camera. An image thresholding operation is employed to transform the image into a black and white bitmap. This bi-leveled lace image is then applied for detecting the borders of the black pixel groups which may be candidates for river segments.

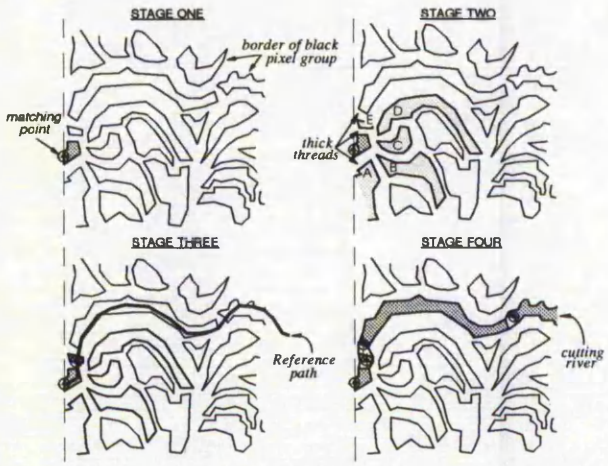


Figure 7: *Using reference path for searching next cutting path*

3) Mapping the reference path into the new frame

Since the lace strip is liable to distort as it is passed through the trimming mechanism, the *reference path* (river) is mapped onto the new frame for the detection of the next cutting river. Some allowance has to be made for cross threads produced as a result of the manufacturing process. These are *thick white threads* which cross the river at intervals and are indistinguishable from the material surrounding the river. For this reason, the detection must allow for small breaks in continuity.

The *LMM technique* has been developed for solving this problem. The detection will be started from the left hand side of the frame and ended at the right. As the matching point has been obtained (described in [3]), the reference path is mapped onto the new frame to find the next border of the river. Several possible connecting borders can be found - A, B, C, D and E labeled in Figure 7 (stage two). The border closest to the mapped reference path is then chosen to become a part of the river (border E is selected in the example). Using the same method, the reference path is repeatedly employed to search the rest of the river segments until it reaches the end of the frame. Once all the segments of the river have been found, lines between adjacent river borders are connected, as illustrated in Figure 7 (stage four), the entire river bank can be constructed. The detailed description of detecting the lace patterns in real-time can be found in [3][4].

3. CORRECTION OF CUTTING OPERATION

A SMP is employed to emulate the movement of the lace strip due to the cutting forces caused by the tactile cutting. A black line is drawn on paper to emulate the river path within a lace strip. The pre-processing vision station captures the image of the paper strip and stores it in the memory. A pattern (desired cutting path) is extracted from the image and transferred to the machine console. This information is used to guide the SMP to draw a second line on the paper strip. Due to the error introduced by the spring, the *path-*

following-errors appear between the desired path and the actual target line. Since the pressure between the SMP and the surface varies every time they come to contact the generated path changes (*path 1, path 2*)

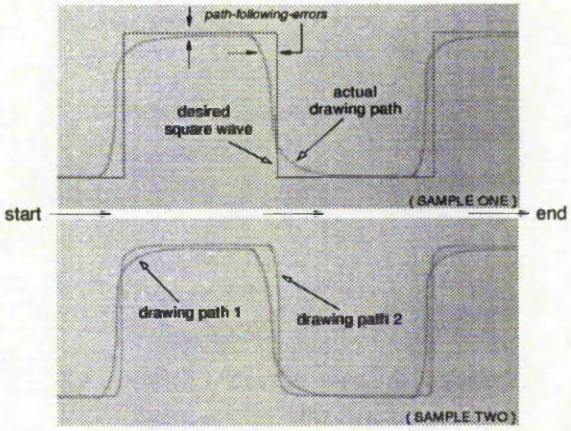


Figure 8: *Samples of square wave following process using the SMP*

A novel method based on modeling human operators' experience and control actions using neural fuzzy theory is developed to solve the complex problems. The intelligent machine console can learn from the experiences (on-line learning) and self-adjust the control actions to match the desired objective. Two A.I. engines are constructed in the system in order to determine a suitable correction added to the original path. The scheme for applying neural fuzzy technique to compensate for the error can be divided into the two functional blocks: *2VMethod*; and *Piecewise Error Compensation Algorithm.*

3.1 The 2VMethod

This approach manipulates a small portion of the intended path to determine a necessary correction for compensating the deviation. Every *two* consecutive coordinates are analysed over the entire path. As a straight line is connected from *Coordinate 1* to *Coordinate 2* (indicated in Figure 9), the angles (θ1 and θ2) between the line (γ) and the X / Y coordinates are used to compute the possible correcting energies *(δ(x) and δ(y))*. The angles θ1 and θ2 are related to the each other - θ1 and θ2 are complementary. These two angles are passed to a *neural fuzzy kernel* which is designed by means of applying neural fuzzy theory to determine a suitable compensation (δ(x) and δ(y)). This correction is separated into two "energies": a) *Correcting pattern;* and b) *Correcting amplitude.*

In the following section, a novel on-line self-learning scheme named the Piecewise Error Compensation Algorithm (PEC Algorithm) based on the neural fuzzy technique derived to calculate the correcting engines is presented.

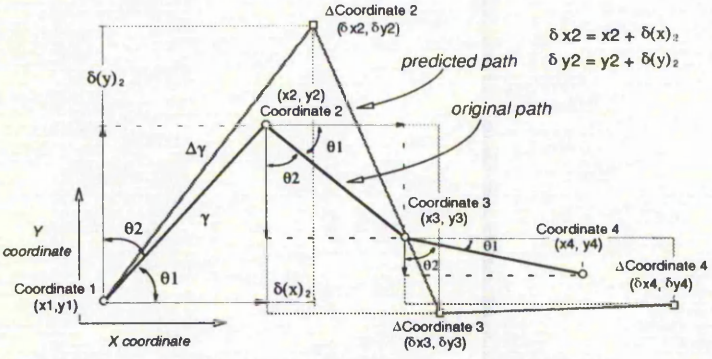


Figure 9: *Calculating new corrected vectors from θ1 and θ2 using the 2VMethod*

3.2 The PEC Algorithm

Two neural fuzzy engines are used by the PEC Algorithm to determine a necessary compensation (correcting pattern and correction amplitude) to eliminate the errors caused by the spring. The neural fuzzy system makes use of neural network for forming the required membership functions and the

rule base. Equation 2 describes the processes of combining the correcting pattern and amplitude to create a compensated path.

$$Predicted\,Segment(i) = Pattern_{segment(i)} \cdot Amplitude_{path} \tag{2}$$

where i is the i^{th} segment of the path. The prediction of the new estimated coordinate $(\Delta Coordinate2(\delta x2, \delta y2))$ is calculated by Equation 3. Equation 4 presents the procedure for computing the pattern of the predicted path.

$$\delta x(i) = x(i) + \delta(x)_i, \quad \delta y(i) = y(i) + \delta(y)_i \tag{3}$$

$$\begin{aligned} Predicted\,Path &= Coordinate(1) + \sum_{i=2}^{n} \Delta Coordinate(i) \\ &= \{x1, y1\} + \sum_{i=2}^{n} \{\delta x(i), \delta y(i)\} \end{aligned} \tag{4}$$

where i is the i^{th} segment of the path and n is the number of coordinates in the path. As the first processed frame is passed under the post-processing vision station, an image is taken and sent to the host system. A software recogniser is used to separate the scanned path and the drawing path (see Figure 10). The top, bottom and centre positions in both paths are taken to measure the inaccuracy of the SMP following process. The distances between these points within the different paths are calculated and passed to the neural fuzzy kernel one which determines a suitable amount of amplitude for the correction. The detailed description of this approach can be found in [5].

4. EXPERIMENTAL RESULTS

A working prototype is constructed. Numerous experiments were carried out to evaluate the efficiency of this scheme. The results of SMP following using the 2VMethod with the PEC Algorithm is illustrated in Figure 11. Through the on-line self-adapting process, the intelligent controller can automatically make the appropriate compensation. According to various experiments, after three frames of correction, almost all the errors caused by the spring can be eliminated. Figure 12 depicts a compensated paths created by the neural fuzzy engine during the self-learning process.

5. CONCLUSION

In the preceding sections of this paper, we have described attempts to develop a vision based intelligent controller using neural fuzzy theory. Just less than three frames of learning process are required before the machine reacts correctly in minimising the errors. According to various experimental results, the developed system can deal with any regular and irregular shapes of path, and

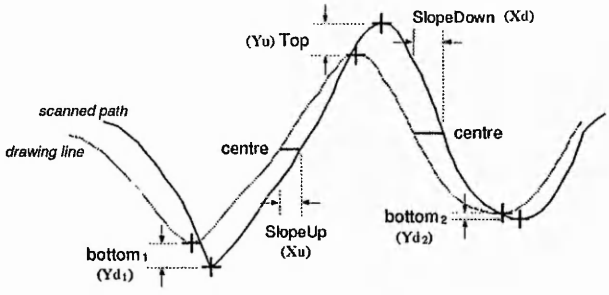


Figure 10: *Detecting the inaccuracy of the SMP following process*

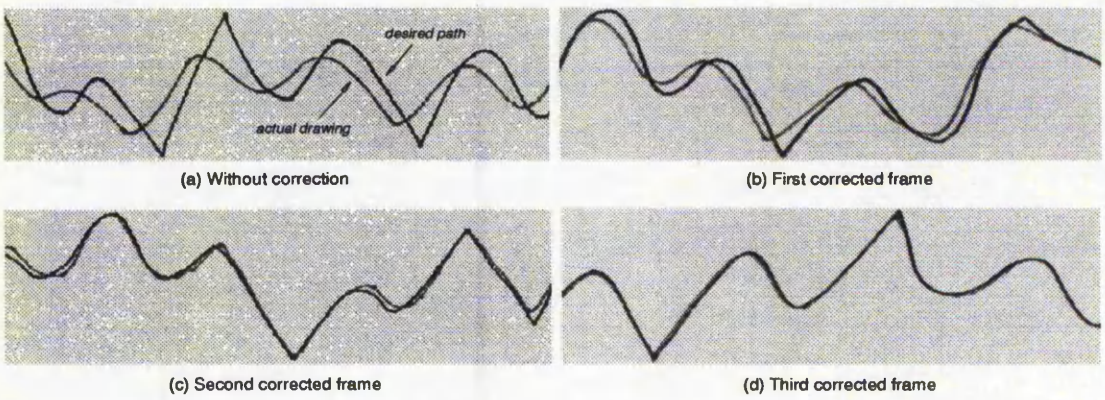


Fig. 11: *Samples of correcting the errors using the 2VMethod and the PEC Algorithm*

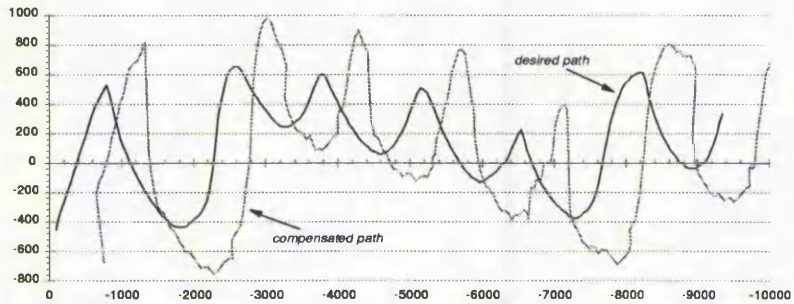


Figure 12: *The detected compensated path*

can produce excellent outcome better than a human operator.

The development of the system is an innovative approach to flexible sheet material processing and has further applications where modeling system behaviour characteristics is difficult. Such systems can range from controlling a robot moving on a slippery surface or piloting a boat. Furthermore, by relying on the intelligent software kernels together with the vision system the controller no longer needs to rely on accurate position feed-back. Backlash, joint flexibility and stick slip can potentially be compensated for by the controller. When characteristics of the mechanism, such as component wear, temperature variation, change over time, the controller can automatically make appropriate compensation.

REFERENCES

[1] King, T.; Tao, L.G.; Jackson, M.R. and Preston, M.E. "Real-time tracking of patterns on deformable materials using DSP", IEE International workshop on systems engineering for real-time applications, Royal Agricultural College, Cirencester, U.K, 1993.

[2] Roberts, D.W. "Analysis of forest succession with fuzzy graph theory", Ecological Modeling, 45:261-274, 1989.

[3] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE Fifth International Conference on Image Processing and Its Applications, Edinburgh, U.K.

[4] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Automation of lace cutting using real-time vision", 8th International Congress on Condition Monitoring and Diagnostic Engineering Management, Queen's University, Canada, 1995.

[5] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Correction of errors due to flexibility of dynamic systems", 1996 IEEE International Conference on Robotics and Automation, Minnesota, USA, 1996.

REAL-TIME TRACKING OF LACE STRETCH USING MACHINE VISION

C-H V. Shih, N. Sherkat, P. Thomas

The Nottingham Trent University, UK

ABSTRACT

This paper describe a computer vision based system for automatic lace scalloping. The main problem other than scalloping path detection in real-time is that of coping with material flexibility. This problem varies depending on the material type and the complexity of the lace pattern. The vision system has to work with many different patterns and sizes of lace as well as tolerating misalignment. In order to satisfy industrial requirements two main conditions must be satisfied. To achieve a sufficient degree of automation, first, the river must be found without prior knowledge of the lace pattern being scalloped. A *Fuzzy Reasoning Rule-based technique* is applied to overcome the problems of material distortion. Next, finding the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach called the *Line Mapping Method (LMM)* is devised to speed up the search for the river in subsequent frames. Several experiments have been carried out using lace patterns of varying complexity. All cutting paths across the patterns were correctly found. Experimental results indicate that the river path can be successfully detected in different lace patterns in real time, while coping with lace distortion.

1. INTRODUCTION

Lace is subject to stretch due to its diaphanous nature. It is mass produced in rolls up to 3.3 meters wide. Traditionally, the strips of lace are separated, manually, along designated paths (river) which run throughout the length of the roll (Figure 1). This is a lengthy and expensive process and results in slowing the rate of production.

A vision based system has been developed for industrial lace cutting. On-line pattern recognition is performed to detect the river. The cutting path is vectorised and transferred to a trimming mechanism. In order to satisfy industrial requirements two main conditions must be satisfied, Sherkat et al (1). To achieve a sufficient degree of automation, first, the river must be found without prior knowledge of the lace pattern scanned. A *Fuzzy Reasoning Rule-based technique* is applied to overcome the problems of material distortion. Next, finding of the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach named the *Line Mapping Method (LMM)* is used to speed up the search for the river in subsequent frames.

Fig. 1: *A typical lace pattern*

Fig. 2: *Bi-level lace bitmap image*

A bi-level image, shown in Figure 2, is used (1). After a thresholding operation a river shows up as a dark area (pixel group) within the edges that cross from one side of the image to the other in a nearly unbroken sequence. There are *thick threads* that cross the river at intervals. These are indistinguishable from the material surrounding the river (marked by circles in Figure 2). Allowance must be made for small breaks in continuity of the river due to these cross threads.

Unlike traditional, rigid engineering materials, lace has essentially no stiffness and can shrink, stretch and distort. To overcome the flexibility problem, we employ an inexact decision making method based on fuzzy rule-based inference technique. As the first cutting river has been recorded, the LMM is engaged to achieve fast detection and higher reliability. The entire system can be broken down into three functional blocks: fuzzy pattern recognition, line mapping process and supervision of the system.

2. FUZZY PATTERN RECOGNITION

Using linguistic terms (variables) the common features of the river shape in various lace patterns are characterised. These common features may be described as:

a) the *position* of the river is around the *centre* of the pattern;
b) the *density* of the river pixel group is not *high*.

The method for applying fuzzy inference techniques to find the first river across the lace pattern with no previous knowledge can be divided into the following tasks:

- Defining system input and output membership functions;
- Fuzzification process;
- Rule evaluation;
- Defuzzification process;
- Verification.

This system reads two input variables *(Group Position and Density)* after each black pixel group has been processed. The fuzzification process sets a value to represent an input's degree of membership in one or more fuzzy variables. During the inference and composition process, strengths are calculated based on antecedent values and assigned to the rules' fuzzy output. Finally, the defuzzification process employs compromising techniques to calculate the average weight for system output. These steps are described in detail as follows.

1) Defining system input and output membership functions

From the linguistic descriptions mentioned above, two system inputs, *group position* and *group density*, can be defined. By monitoring the position and density of the black pixel groups (Figure 2) across a lace pattern, a fuzzy inference engine can determine whether the pixel group is a possible segment of a river.

Two initial experiments were carried out to define the fuzzy input and output membership functions. Frequency histograms were used on the sample data to define input membership functions, Roberts (2) and Turksen (3). In this way can be obtained a set of data from the *River group* part to build the membership functions.

2) Fuzzification and rule evaluation

The fuzzification process computes an input value to represent a degree of membership in one or more fuzzy variables. These fuzzified inputs are processed through a pre-defined set of rules using min-max evaluation to form fuzzified outputs. The Fuzzy Associative Memory Bank is applied here to reduce the number of the rules for speeding up the calculation.

3) Defuzzification process

Defuzzification process is to convert its fuzzy outputs into a single raw or crisp output. In these experiments, we choose the *"centre-of-gravity method"* for resolving both the vagueness and conflict issues, Zadeh (4). The weighted average is calculated as follows:

$$Weighted\ average = \frac{\sum(shaded\ area \times centroid\ point)}{\sum(shaded\ area)}$$

By relying on the use of fuzzy inference technique, each black pixel group could be calculated and assigned an average weight (*possibility*).

4) Verification

Once *all* the black pixel groups have been assigned a *possibility value*, the pixel groups whose possibility values are less than 50% are abandoned. The verification process can then be broken down into the following tasks:

a) Calculate the distance between two adjacent groups;
b) If the distance is shorter than a specified value (set to six pixels long in these experiments) a network is built to record this path;
c) Continuously trace the distances between pixel groups while recording all the correct paths until a new pixel group reaches the border of the image (right hand edge of the frame);
d) Calculate the total possibility values and divide by the number of the group in this path (*average possibility*);
e) If the *average possibility* is bigger than a specified value, (75% was used in the experiments) then the correct river has been found; if the average possibility is less than this value, repeat step (c) to (e) until the correct river is located.

By calculating the distances and tracing the average possibilities in all these segments, the river, highlighted in Figure 3, can be located. The detailed description of the fuzzy pattern recognition system can be found in Sherkat et al (5)(6).

3. LINE MAPPING PROCESS

When the first cutting river in the lace strip is successfully detected, the extracted knowledge can be used to speed up the search in subsequent frames. In order to meet the real-time requirements of the system, instead of using traditional pattern matching techniques, a new approach called the *Line Mapping Method (LMM)* has been developed to achieve fast response and higher reliability. This approach is divided into the following processes:

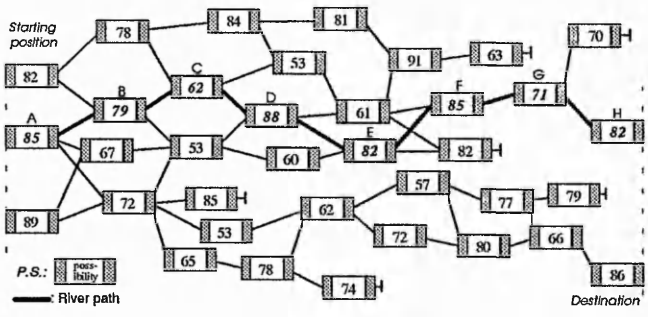


Fig. 3: *Interconnection between river segments*

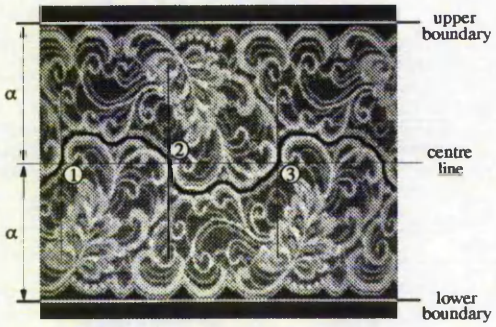


Fig. 4: *Extracting a repeat cutting cycle*

1) Indicating and registering one repeat cutting cycle

A *centre line* is located by calculating the distance between the *upper* and *lower boundaries* shown in Figure 4. Three crossing points between the cutting river and the centre line are marked. The cutting path (river) between the intersections ① and ③ labeled in Figure 4 indicates a repeat cutting cycle, which acts as a *reference path* for detection of subsequent frames.

2) Capturing the following frame

The next frame of a 256 grey scale lace image is captured by the frame grabber from the CCD camera and temporarily stored in a memory block. An image thresholding operation is employed to transform the image into a black and white bitmap. This bi-leveled lace image is then applied for detecting the borders of the black pixel groups which may be candidates for river segments. As depicted in Figure 5, the border following technique is used to find the borders (outlines) of the potential river segments.

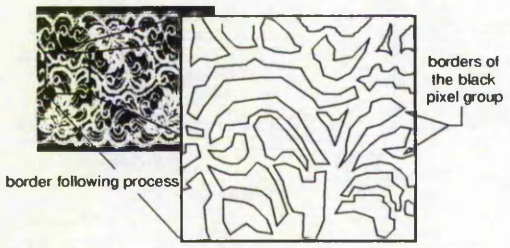


Fig. 5: *Borders of the black pixel group*

3) Mapping the reference path into the new frame

Since the lace strip is liable to distort as it is passed through the trimming mechanism, the *reference path* (river) is mapped onto the new frame for the detection of the next cutting river. With careful inspection it is clear, from Figure 6, that the two halves of the image do not completely match (the *reference path* is not completely within the river banks).

A river, as stated previously, crosses from one side of the image to the other in a nearly unbroken sequence. Some allowance has to be made for cross threads produced as a

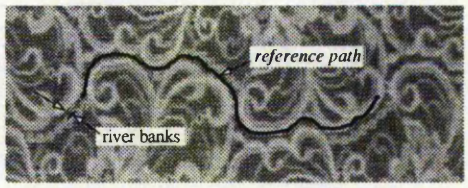


Fig. 6: *Mapping the reference path*

result of the manufacturing process. These are *thick white threads* which cross the river at intervals and are indistinguishable from the material surrounding the river. For this reason, the detection must allow for small breaks in continuity.

The *LMM technique* has been developed for solving this problem. The detection will be started from the left hand side of the frame and ended at the right. As the matching point has been obtained (described in Section 4), the reference path is mapped onto the new frame to find the next border of the river. Several possible connecting borders can be found - A, B, C, D and E labeled in Figure 7 (stage two). The border closest to the mapped reference path is then chosen to become a part of the river (border E is selected in the example). Using the same method, the reference path is repeatedly employed to search the rest of the river segments until it reaches the end of the frame. After all the segments of the river have been found, lines between adjacent river borders are connected, as illustrated in Figure 7 (stage four), the entire river bank can be constructed.

4. SUPERVISION OF THE SYSTEM

Since the CCD camera is mounted on the X axis of the machine, the camera is moved with the cutter. The advantage of using such a construction is that the camera and the cutter are kept in a constant position relative to each other. Consequently, it is easy to calculate the real cutting position from the captured image, as well as to correct the errors between these two captured frames. On the other hand, since the lace strip is transported past the

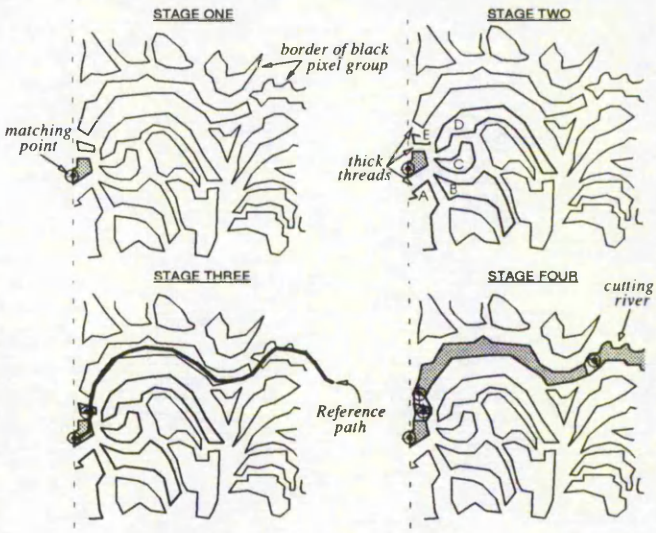


Fig. 7: *Searching a new cutting path*

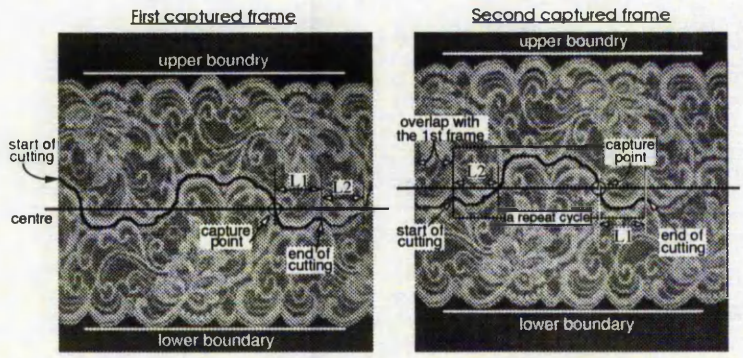


Fig. 8: *Vision and cutting procedure*

vision system by the conveyor belt following the Y axis, the vision system has to consider the more complex two dimensional image shifting problem. Nevertheless, this *"look-and-move"* strategy yields more accurate results than the *"eye-to-hand co-ordination"* approach, Wolfe et al (7), and also avoids small drift due to material length or missing steps of the motor(s). The strategy for analysing images moving in two directions and generating the vector data for machine control is discussed in the following sections.

4.1 Detecting the first river and finding the capture point for next frame

As the lace strip is transported past the field of vision, the first frame of the lace image is captured and temporarily stored in memory. After the fuzzy reasoning process, the cutting rivers across the lace pattern are found. The next stage of the system will then decide the *capture point* on the cutting path for the second lace image. When the machine is in operation, the camera is moving together with the cutter, so finding the position where the camera can capture a similar image for the *LMM* process is critical. As shown in Figure 8, a centre line can be drawn across the first frame, and an intersection between the cutting river can then be found. This position is engaged for grabbing the second frame of the lace image.

4.2 Generating machine movement data and grabbing the second frame

While the first cutting river has been detected, using the fuzzy reasoning method, the machine control data is generated and downloaded to the machine controller. The controller transforms the motion data into the real machine movement data and starts driving the cutter to cut the strip of lace. When the machine starts cutting the lace strip, the controller simultaneously responds to the machine console with the current position on the XY axes. The machine console then continuously tracks the cutting positions until it reaches the *capture point* (shown in Figure 8 - first captured frame). Consequently, the CCD camera is triggered to capture the second frame of the lace image which is stored into memory for processing.

Since the system takes approximately two hundred milli-seconds to find the next cutting river, this will stop the cutting process between two captured frames. To solve this problem, we simply add a quarter of the repeat cutting cycle (*L1*, between *capture* and *end of cutting points*, indicated in Figure 8) to the cutting path. Thus, while the machine is trimming past the *capture point*, the vision system grabs a frame as well as finding the cutting river before the machine actually ends trimming. This enables continuous operation of the system in real-time.

4.3 Finding the reference path and the next capture point

As the second lace image is stored in the memory, the fuzzy reasoning rule-based technique is, again, employed to find the second frame of the lace image. The second intersection with the cutting river can be designated as the *capture point* for the next frame. As the machine continuously trims the lace and reaches the *'end of cutting'* position in the first frame, the movement data of the second cutting path has already been produced and stored. Therefore, the machine could continuously cut the lace pattern through subsequent frames in an unbroken sequence.

L1 and L2 indicated in Figure 8 are taken from the first frame, and coupled to the second frame for determining the length and shape of the reference path (a repeat cutting cycle). After the reference path has been defined, its corresponding position with the centre line (first pixel of this module) is then registered. This will be used for detection of subsequent frames.

4.4 Line mapping operation

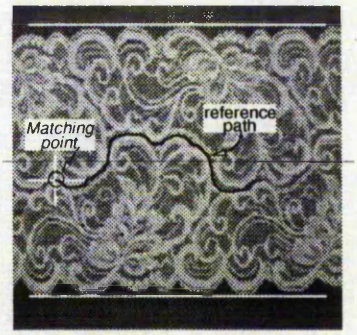


Fig. 9: *Mapping the reference path*

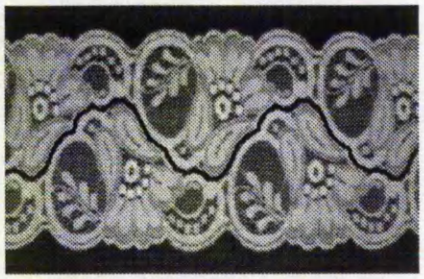


Fig. 10: *An example of river extraction*

After the *reference path* has been determined, the extracted knowledge can be used to speed up the search for the river in subsequent frames. Figure 9 shows an example of mapping the reference path into the following frame. Utilising the *Line Mapping Method (LMM)*, the new cutting river across the lace image can be successfully and quickly detected.

5. EXPERIMENTAL RESULTS

Various experiments were carried out to investigate the effectiveness of this method. Numerous lace patterns were employed for detecting the river location. All cutting paths across the patterns were successfully found. The time taken to isolate the river and produce cutting path depends on complexity of the pattern. Time taken for most kinds of motif, using the fuzzy reasoning rule-based technique, is typically about 300 milli-seconds using an Intel 80486 processor running at 66 MHz. Nevertheless, in the case of a very few intricate lace patterns (e.g. Figure 1), up to 1.5 seconds is required. Once the river path on the first frame is found, this knowledge can be utilised by the LMM to speed up the detection for the river in subsequent frames in real time. The time to detect a repeat cutting path using LMM is dependent on how complex the motif is, the length of one repeat cutting cycle and the distortion of the pattern. On most kind of lace patterns detection time is about 150 to 200 milli-seconds. The frame grabber digitises a incoming video signal at a rate of 30 frames per second. Typically a repeat cutting cycle of the lace strip is around 9 to 15 cm. Therefore the speed for tracking the lace pattern using the LMM is approximately 25 to 35 meters / minute.

The speed for searching the river location across lace strip using LMM is appropriately three to five times faster than using the fuzzy detecting method alone. A sample lace pattern together with the resulting river path is shown in Figure 10. Also a distorted lace sample with its successfully detected river is depicted in Figure 11.

6. CONCLUSION

In the preceding sections of this paper, we have described attempts to develop a *fuzzy reasoning rule-based system* and the *line mapping method* for detecting varied types of lace patterns in real time. Experimental results indicate that the objectives have mostly been fulfilled. The system requires no prior knowledge of any particular lace pattern or any training. According to the results of the

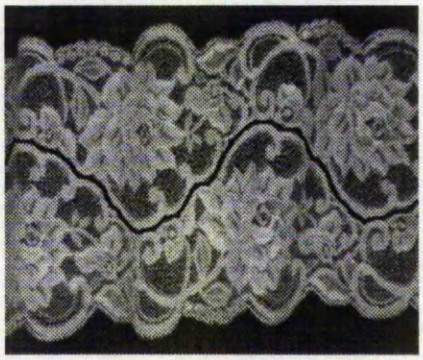


Fig. 11: *Lace pattern under 40% contraction (successfully detected)*

experiments, a combination of fuzzy pattern recognition technique and the LMM can be sued to successfully detect the cutting river within various lace patterns in real time. Compared with the conventional image processing methods, (1), Russell and Wong (8), it is not only easier to design and implement the system, but also more effective in coping with distortion. Furthermore it does not require any training or prior knowledge of the lace pattern.

REFERENCES

1. Sherkat, Nasser; Birch, Mike and Thomas, Peter. "Real-time vision for automatic lace cutting", IS&T/SPIE Symposium on Electronic Imaging Science & Technology, 6-10 February 1994, pp.322-333.
2. Roberts, D.W. "Analysis of forest succession with fuzzy graph theory", Ecological Modeling, 45:261-274, 1989.
3. Turksen, I.B. "Measurement of fuzziness: interpretation of the axioms of measure", Proceeding of the Conference on Fuzzy Information and Knowledge Representation for Decision Analysis, IFAC, Oxford, 1984, pp.97-102.
4. Zadeh, Lofti A. "Fuzzy logic", Computer, IEEE, April 1988, pp.83 - 93.
5. Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Application, kawasaki, Japan, Dec. 13-15, 1994, pp.61-64.
6. Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology, California, USA, Feb. 5-10, 1995.
7. Wolfe, D.F.H.; Wijesoma, S.W. and Richards, R.J. "Eye-to-hand co-ordination", Assembly Automation, Vol.11, No.1, 1991, pp.15-20.
8. Russell, R.A. and Wong, P. "Automation of lace cutting using computer vision", Robots: Coming of Age. Proceedings of International Symposium and Exposition on Robots. Designed the 19th ISIR by the International Federation of Robotics, Nov. 1988, pp.385-93.

AUTOMATION OF LACE CUTTING USING REAL-TIME VISION

Chi-Hsien V. SHIH, Nasser Sherkat, Peter Thomas

Department of Computing, The Nottingham Trent University,
Burton Street, Nottingham, UK, NG1 4BU

ABSTRACT

During the last few years automating lace handling has received much attention in the United Kingdom. Research and development is ongoing in the two main areas of automatic lace cutting and automatic lace quality inspection. This paper reports on research and development work under gone at the Nottingham Trent University, in collaboration with local industry, in automation of lace scalloping. The research is reviewed in its various stages of development. Three different methods together with their associated experimental results are described and the merits of each are discussed. An innovative approach, based on neural fuzzy logic, for dealing with the problem of lace distortion, due to its flexible nature, is described.

1. INTRODUCTION

Lace, intended for decoration of clothing and furniture, is mass produced in wide rolls. In order to increase the production rate, the rolls are made by repeating the desired lace strips side-by-side. At the second manufacturing stage the repeated strips are separated along pre-designated paths (rivers) which run throughout the length of the roll (Figure 1). The process of trimming the lace has been carried out manually. This is a lengthy and expensive process and results in slowing the rate of production. A small number of machines have been developed that use a simple passive cutter mechanism that relies on the structural strength of the lace pattern. In such systems the cutter is held stationary while the lace is run against it. This approach is only suitable in cases where the lace pattern is of very shallow scallop. In the case of deep scallop patterns a more sophisticated method of guidance, based on an active cutter, is required.

Figure 1: *A typical lace pattern*

A vision based system has been developed (Figure 2) for industrial lace cutting. On-line pattern recognition is performed to detect the river. The cutting path is vectorised and transferred to a trimming mechanism. In order to satisfy industrial requirements two main conditions must be satisfied [1]. To achieve a sufficient degree of automation, first, the river must be found without prior knowledge of the lace pattern scanned. Two attempts which applied *traditional image processing methods* and *fuzzy pattern recognition technique* have been made to detect varied shapes of rivers within the lace patterns. Next, finding of the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach called the *Line Mapping Method (LMM)* is used to speed up the search for the river in subsequent frames.

Work has been reported in using lasers to cut deformable materials [2]. Although using lasers reduces this deformation, distortion due to mechanical feed misalignments persists. In order to tackle the problem of distortion due to material flexibility in general, a novel approach has been developed and described here. A *Spring Mounted Pen (SMP)* is used in the experiments to emulate the distortion of lace pattern caused by the tactile cutter. Using the pre- and post-processing vision systems, it is possible to monitor the scalloping process as well as generating on-line information to the host system. This allows overcoming the problems of lace distortion due to the trimming operation. This paper is structured into three sections: detection of first cutting river, line mapping process, and correction of cutting operation.

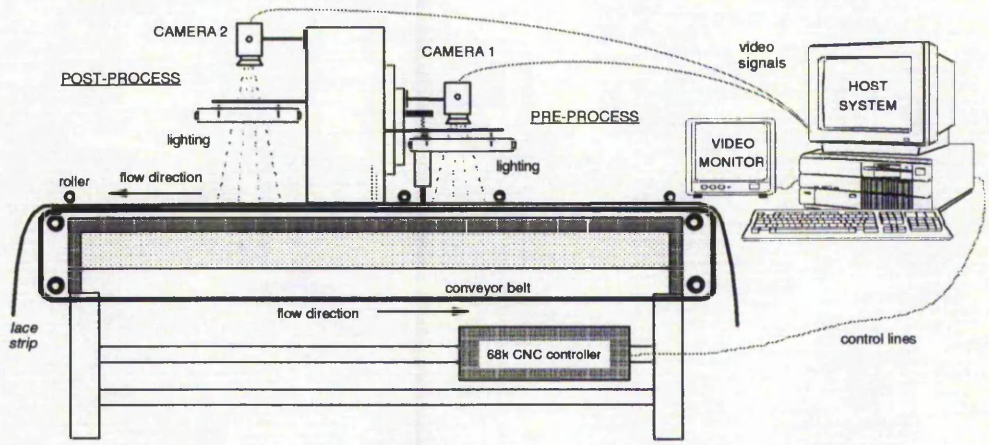


Figure 2: *Schematic of the vision based control system*

2. DETECTION OF FIRST CUTTING RIVER

Two attempts have been made to find the first river within the lace pattern without prior knowledge of the pattern scanned.

The first scheme [1] is based on pixel intensity directed feature extraction. In order to detect the river, a pixel intensity map is created to determine significant differences. This approach hinges on detection of large variations in intensity to highlight the river. Other contextual information such as pattern repeat cycle, river continuity and contour closing is used to speed up the process of feature extraction. This process can be broken down into the following tasks:

1) Edge finding

By finding the average intensity of each vertical strip of the lace image and examining which points lie above and below this threshold a clear pattern emerges with most of the area within the edges being brighter and all of the area outside the edges being darker. Thus, by finding the first and last points in a vertical strip which are above this value, the edge of the lace at that position is found. The points above and below the threshold are stored as 1 and 0 respectively in a bitmap (Figure 3).

2) Finding the pattern repeat

The technique used here is based on looking for 'landmarks' within the bit map, the most prominent being the large dark areas within the pattern. By finding the distance between each dark area and all subsequent ones, the most frequently occurring distance will be due to the pattern repeating.

3) Finding the river candidates

Taking a column down one side of the image it would be possible, excluding cross threads, to simply take each dark area and follow it to the right until:

- the other side is reached;
- its boundary left the edge of the lace;
- it could proceed no further.

This would be marginally more complex if the river was allowed to loop back on itself but because of manufacturing techniques for the type of application considered here, this does not happen. As each river stretches along the length of the image, its top and bottom edge is stored. If this becomes too wide then the river can be discarded since this must be part of the pattern. When the river can proceed no further by these means, it must initially be assumed that it has reached a cross thread. If this is not the initial search, then the previous river can be used to indicate the

Figure 3: *Bi-level lace bitmap image*

direction of the next point at the current stage of the pattern. However, for the first river the system must look for the nearest dark point without backtracking. A bias can be placed on this search depending on the direction of the previous points and distance from the middle. If the distance to this point is greater than a pre-determined threshold then the river can be considered to have reached a premature end and is considered to be invalid.

4) Finding the correct river from this list.

The rivers found in the previous step can be tested against each other, as they are found, and only maintain the best case. Two values are required to compare every two rivers. The first is a measure of symmetry within the width of the lace. The second is a measure of repeatability. By comparing these two values, for each river, the best alternative can be found. When the best river has been found, the repeatability and symmetry values can be compared with a pre-determined threshold to ensure that it is satisfactory. If not, the machine must stop rather than cut a wrong path.

The detection of the river heavily depends on the feature of the repeated cutting path. The two extremes of the river should be equi-distant from their nearest edge, and after a distance equal to the repeat 'period' of the design. The river should be back at the same position relative to the two edges as it was when it started. As the lace pattern is distorted, these features of the river are no longer presented. This causes the dissatisfied results when the system applies this scheme to analyse the lace pattern contained the distortion bigger than 5-10 percents.

In order to overcome the problems of material distortion due to the transporting process, a *Fuzzy Reasoning Rule-Based Technique* is presented in the system. This approach can be divided into the following functional blocks:

- Defining system input and output membership functions;
- Fuzzification process;
- Rule evaluation;
- Defuzzification process;
- Verification.

1) Defining system input and output membership functions

After a thresholding operation a river shows up as a dark area (pixel group) within the edges that cross from one side of the image to the other in a nearly unbroken sequence. There are *thick threads* that cross the river at intervals. These are indistinguishable from the material surrounding the river (marked by circles in Figure 3). Allowance must be made for small breaks in continuity of the river due to these cross threads.

The objective here is to find the river along a lace pattern, by using linguistic variables to represent the common feature of the river shape in various lace patterns. These common features may be described as:

a) the *position* of the river is around the *centre* of the pattern;
b) the *density* of the river pixel group is not *high*.

From the linguistic descriptions, two system inputs, *group position* and *group density*, can be defined. By monitoring the position and density of the black pixel groups across a lace pattern, a fuzzy inference engine can determine whether the pixel group is a possible segment of a river.

Two initial experiments were carried out to define the fuzzy input and output membership functions. Frequency histograms were used on the sample data to define input membership functions [3][4]. In this way a set of data can be obtained from the *River group* part to build the membership functions.

Rule (A): IF Position is Right and Density is P.S.
THEN Possibility is Negative Medium
Rule (B): IF Position is Near Mid. and Density is N.S.
THEN Possibility is Positive Medium

Poss-ibility		Position				
		Left	ML	Mid	MR	Right
Density	NL	NM	PS	PS	PS	NM
	NS	PS	PM (B)	PM	PM	PS
	Med	PS	PL	PL	PL	PS
	PS	NM	Med	PS	Med	NM (A)
	PL	NL	NL	NL	NL	NL

Figure 4: *Fuzzy Associative Memory (FAM) Bank to determine the possibility*

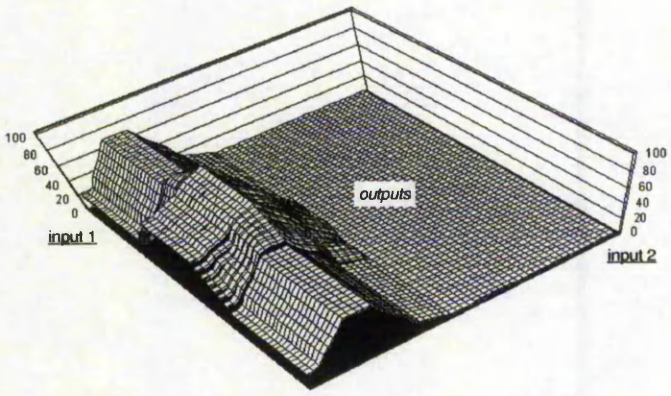


Figure 5: *Output pattern of the fuzzy engine*

2) Fuzzification and rule evaluation

The fuzzification process computes an input value to represent a degree of membership in one or more fuzzy variables. These fuzzified inputs are processed through a pre-defined set of rules using min-max evaluation to form fuzzified outputs. As depicted in Figure 4, the Fuzzy Associative Memory Bank is applied here to reduce the number of the rules for speeding up the calculation.

3) Defuzzification process

Defuzzification process is to convert its fuzzy outputs into a single raw or crisp output. In these experiments, we choose the *"centre-of-gravity method"* for resolving both the vagueness and conflict issues [5]. The weighted average is calculated as follows:

$$Weighted\ average = \frac{\sum(shaded\ area \times centroid\ point)}{\sum(shaded\ area)}$$

By relying on the use of fuzzy inference technique, each black pixel group could be calculated and assigned an average weight (*possibility*). Figure 5 illustrates the output pattern of the inference engine.

4) Verification

Once *all* the black pixel groups have been assigned a *possibility value*, the pixel groups whose possibility values are less than 50% are abandoned. The verification process can then be broken down into the following tasks:

a) Calculate the distance between two adjacent groups;
b) If the distance is shorter than a specified value (set to six pixels long in these experiments) a network is built to record this path;
c) Continuously trace the distances between pixel groups while recording all the correct paths until a new pixel group reaches the border of the image (right hand edge of the frame);
d) Calculate the total possibility values and divide by the number of the group in this path (*average possibility*);
e) If the *average possibility* is bigger than a specified value, (75% was used in the experiments) then the correct river has been found; if the average possibility is less than this value, repeat step (c) to (e) until the correct river is located.

By calculating the distances and tracing the average possibilities in all these segments, the river, highlighted in Figure 6, can be located. The detailed description of the fuzzy pattern recognition system can be found in [6][7].

3. LINE MAPPING PROCESS

When the first cutting river in the lace strip is

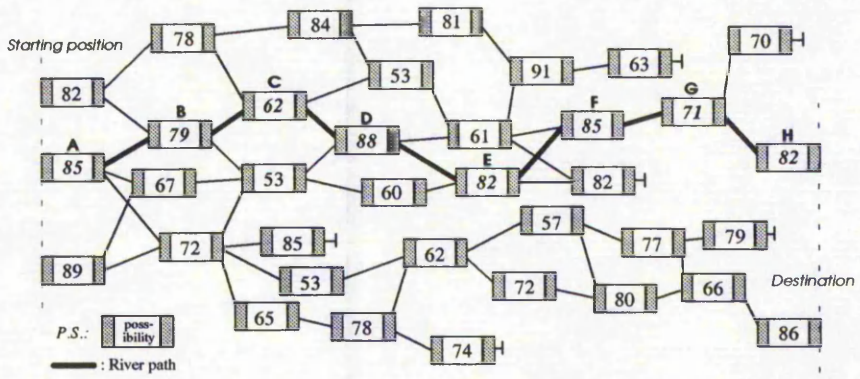


Figure 6: *Interconnection between each possible river segments*

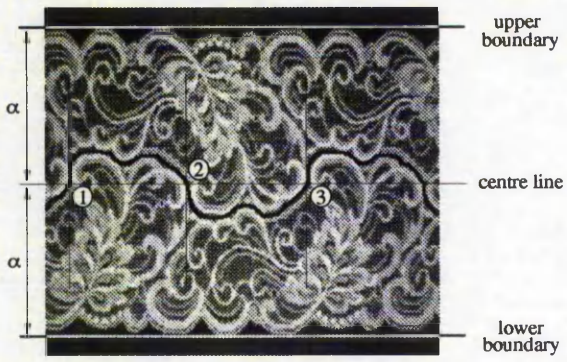


Figure 7: *Extracting a repeat cutting cycle*

successfully detected, the extracted knowledge can be used to speed up the search in subsequent frames. In order to meet the real-time requirements of the system, instead of using traditional pattern matching techniques, a new approach called the *Line Mapping Method (LMM)* has been developed to achieve fast response and higher reliability. This approach is divided into the following processes:

1) Indicating and registering one repeat cutting cycle

A *centre line* is located by calculating the distance between the *upper* and *lower boundaries* shown in Figure 7. Three crossing points between the cutting river and the centre line are marked. The cutting path (river) between the intersections ① and ③ labeled in Figure 7 indicates a repeat cutting cycle, which acts as a *reference path* for detection of subsequent frames.

2) Capturing the following frame

The next frame of a 256 grey scale lace image is captured by the frame grabber from the CCD camera and temporarily stored in a memory block. An image thresholding operation is employed to transform the image into a black and white bitmap. This bi-leveled lace image is then applied for detecting the borders of the black pixel groups which may be candidates for river segments. As depicted in Figure 8, the border following technique is used to find the borders (outlines) of the potential river segments.

3) Mapping the reference path into the new frame

Since the lace strip is liable to distort as it is passed through the trimming mechanism, the *reference path* (river) is mapped onto the new frame for the detection of the next cutting river. With careful inspection it is

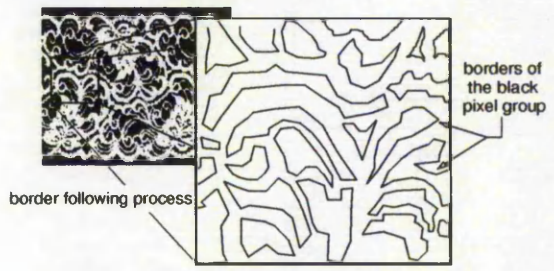


Figure 8: *Borders of the black pixel group*

clear, from Figure 9, that the two halves of the image do not completely match (the *reference path* is not completely within the river banks).

A river, as stated previously, crosses from one side of the image to the other in a nearly unbroken sequence. Some allowance has to be made for cross threads produced as a result of the manufacturing process. These are *thick white threads* which cross the river at intervals and are indistinguishable from the material surrounding the river. For this reason, the detection must allow for small breaks in continuity.

The *LMM technique* has been developed for solving this problem. The detection will be started from the left hand side of the frame and ended at the right. As the matching point has been obtained (described in [8]), the reference path is mapped onto the new frame to find the next border of the river. Several possible connecting borders can be found - A, B, C, D and E labeled in Figure 10 (stage two). The border closest to the mapped reference path is then chosen to become a part of the river (border E is selected in the example). Using the same method, the reference path is repeatedly employed to search the rest of the river segments until it reaches the end of the frame. After all the segments of the river have been found, lines between adjacent river borders are connected, as illustrated in Figure 10 (stage four), the entire river bank can be constructed.

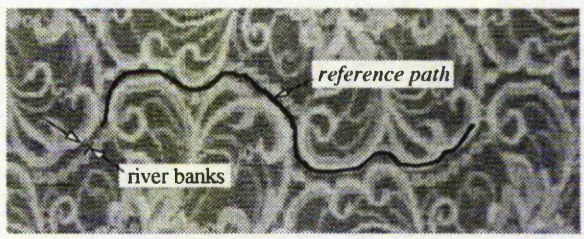


Figure 9: *Mapping the reference path into a new lace image*

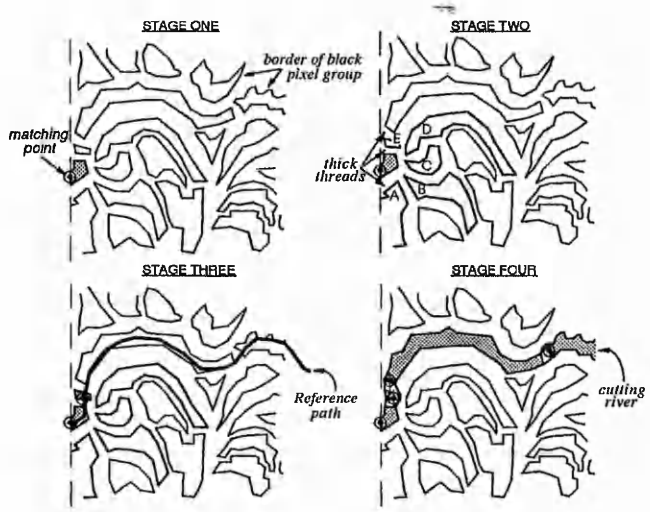


Figure 10: *Using reference path for searching next cutting path*

4. CORRECTION OF CUTTING OPERATION

As depicted in Figure 11, a spring is used to connect between a pen and the Z axis of the cutting mechanism (*Spring Mounted Pen, SMP*). This device is employed to emulate the movement of the lace strip due to the cutting forces caused by the tactile cutting.

A black line is drawn on paper to emulate the river path within a lace strip. The pre-processing vision system (see Figure 2) captures the image of the paper strip and stores in the memory. A pattern (desired cutting path) is extracted from the image and transferred to the machine console. This information is used to guide the SMP to draw a second line on the paper strip. Figure 12 represents the results of following a square wave. Due to the imperfection generated by the spring, the *path-following-errors* appear between the desired path and the actual target line. As the SMP is born different pressure, the pattern of the SMP drawing will alter (*path 1* and *path 2* indicated in Figure 12).

A novel method based on modeling human operators' experience and control actions using inexact algorithm, e.g. fuzzy logic, neural networks, and neural fuzzy technique, is developed to solve the complex problems. The intelligent machine console can learn from the experiences (on-line learning) and self-adjust the control actions to match the desired objective. Figure 13 shows the data flow diagram for the system overview. Two A.I. engines are constructed in the system in order to determine a suitable correction added to the original path. The system can deal with any irregular shapes of paths. According to various experimental results, the machine applied this method can produce the excellent outcome better than a human operator. The detailed description of this approach is beyond the scope of this paper.

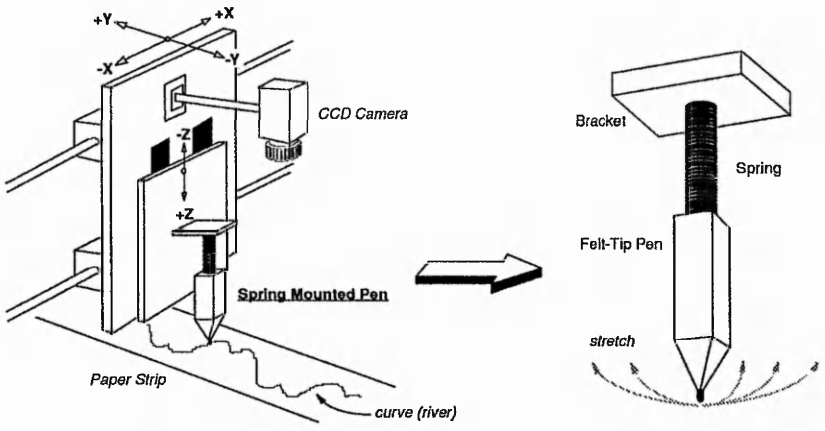


Figure 11: *Spring Mounted Pen (SMP) connected with the testing rig*

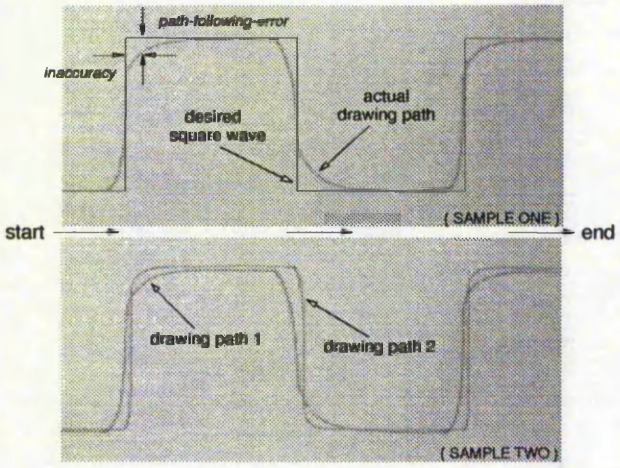


Figure 12: *Samples of square wave following process*

5. EXPERIMENTAL RESULTS

A number of experiments were carried out to evaluate the effectiveness of this method. Numerous lace patterns were employed for detecting the river location. All cutting paths across the patterns were successfully found. The time taken to isolate the river and produce cutting path depends on complexity of the pattern. Time taken for most kinds of motif, using the fuzzy reasoning rule-based technique, is typically about 300 milli-seconds using an Intel 80486 processor running at 66 MHz. Nevertheless, in the case of a very few intricate lace patterns, up to 1.5 seconds is required. Once the river path on the first frame is found, this knowledge can be utilised by the LMM to speed up the detection for the river in subsequent frames in real time. The time to detect a repeat cutting path using LMM is dependent on how complex the motif is, the length of one repeat cutting cycle and the distortion of the pattern. On most kind of lace patterns detection time is about 150 to 200 milli-seconds. The frame grabber digitises a incoming video signal at a rate of 30 frames per second. Typically a repeat cutting cycle of the lace strip is around 9 to 15 cm. Therefore the speed for tracking the lace pattern using the LMM is approximately 25 to 35 meters / minute. Two sample lace patterns together with the resulting river paths are shown in Figure 14 and Figure 15. Also a distorted lace sample with its successfully detected river is depicted in Figure 16.

6. CONCLUSIONS

The objective of detecting the river in an unseen lace pattern in real-time has been achieved. This has enabled the development of a working prototype for an automatic lace scalloping machine. It is found that the biggest problem in automating the process of lace scalloping is that of dealing with lace distortion in real time. Distortion, not only creates problems for the pattern recognition task, it also complicates the feeding and cutting processes.

Compared with the approaches used traditional image processing methods [1] and the fuzzy pattern recognition technique to find the first river without prior knowledge, the fuzzy logic based approach is more effective. Applying the fuzzy technique the river in the lace pattern up to 40 % contraction is successfully detected where the traditional method is

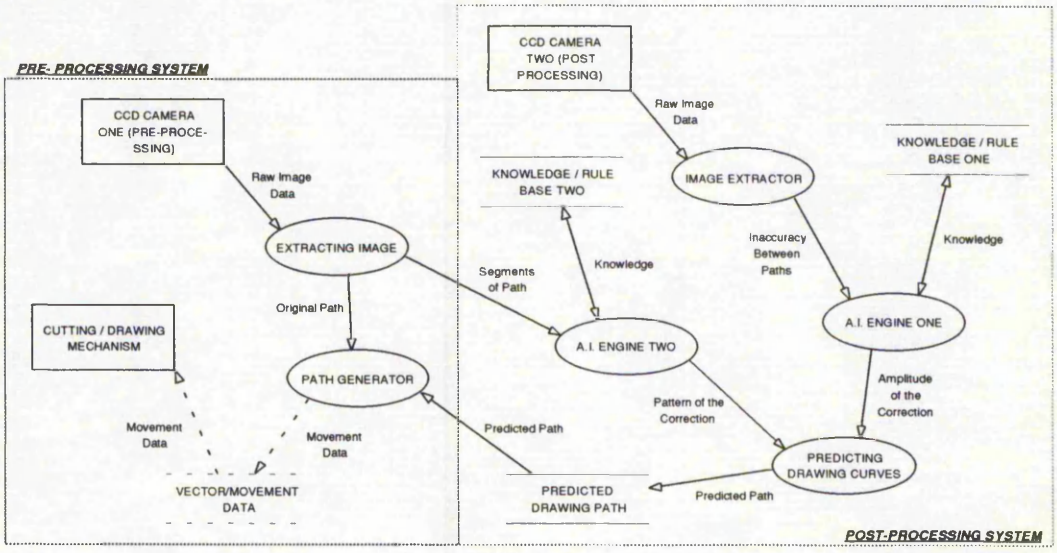


Figure 13: *The overview of the vision and motion control systems*

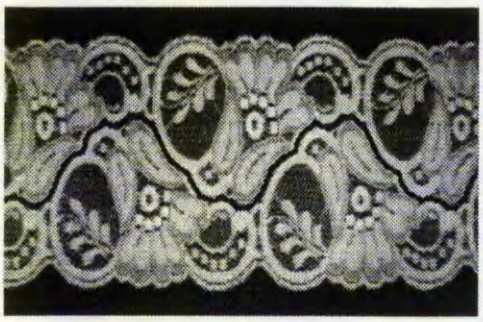


Figure 14: Example A of river extraction

Figure 15: Example B of river extraction

failed. According to the results of the experiments, a combination of fuzzy pattern recognition technique and the LMM can be applied to detect the distorted river within various lace patterns in real time. Besides, in contrast to the scheme mentioned in [9], the proposed algorithm is not only easier to design and implement, but also more effective in coping with distortion. Furthermore it does not require any training or prior knowledge of the lace pattern.

7. ACKNOWLEDGMENTS

This work has been carried out in collaboration with PACER Systems Ltd. Unit 6, Robin Hood Industrial Estate, Nottingham, NG3 1GE, UK.

8. REFERENCES

[1] Sherkat, Nasser; Birch, Mike and Thomas, Peter. "Real-time vision for automatic lace cutting", IS&T/SPIE Symposium on Electronic Imaging Science & Technology, 6-10 February 1994, pp.322-333.

[2] T. King, "Real-time tracking of patterns on deformable materials using DSP," IEE International workshop on systems engineering for real-time applications, Royal Agricultural College, Cirencester, U.K, 13-14 September 1993.

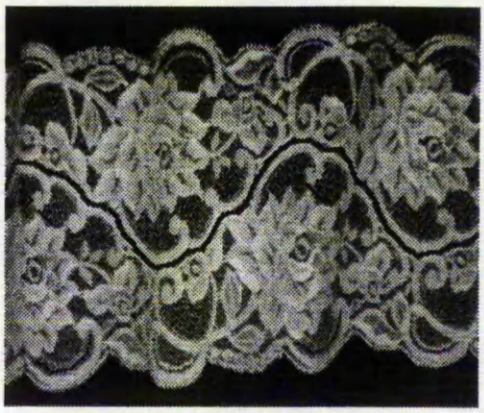


Figure 16: Lace pattern under 40% contraction (successfully detected)

[3] Roberts, D.W. "Analysis of forest succession with fuzzy graph theory", Ecological Modeling, 45:261-274, 1989.

[4] Turksen, I.B. "Measurement of fuzziness: interpretation of the axioms of measure", Proceeding of the Conference on Fuzzy Information and Knowledge Representation for Decision Analysis, IFAC, Oxford, 1984, pp.97-102.

[5] Zadeh, Lofti A. "Fuzzy logic", Computer, IEEE, April 1988, pp.83 - 93.

[6] Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Application, Kawasaki, Japan, Dec. 13-15, 1994, pp.61-64.

[7] Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology, California, USA, Feb. 5-10, 1995.

[8] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE Fifth International Conference on Image Processing and Its Applications, Edinburgh, U.K.

[9] Russell, R.A. and Wong, P. "Automation of lace cutting using computer vision", Robots: Coming of Age, Proceedings of International Symposium and Exposition on Robots, Designed the 19th ISIR by the International Federation of Robotics, Nov. 1988, pp.385-93.

Close coupling of pre- and post-processing vision stations using Inexact Algorithms

Chi-Hsien V. Shih, Nasser Sherkat, Peter Thomas

The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom

ABSTRACT

Work has been reported using lasers to cut deformable materials. Although the use of laser reduces material deformation, distortion due to mechanical feed misalignment persists. Changes in the lace pattern are also caused by the release of tension in the lace structure as it is cut. To tackle the problem of distortion due to material flexibility, the *2VMethod* together with the *Piecewise Error Compensation Algorithm* incorporating the *inexact algorithms*, i.e., *fuzzy logic*, *neural networks* and *neural fuzzy technique*, are developed. A *Spring Mounted Pen* is used to emulate the distortion of the lace pattern caused by tactile cutting and feed misalignment. Using pre- and post-processing vision systems, it is possible to monitor the scalloping process and generate on-line information for the Artificial Intelligence engines. This overcomes the problems of lace distortion due to the trimming process. Applying the algorithms developed, the system can produce excellent results, much better than a human operator.

Keywords: machine vision, pre- and post-processing vision stations, self-learning, fuzzy logic, neural networks, neural fuzzy, Inexact Algorithms, PEC Algorithm, 2VMethod, Spring Mounted Pen.

1. INTRODUCTION

A number of attempts have been made to automate the process of lace scalloping and quality inspection[1,2,3,4,5]. Work has been reported in using lasers to cut deformable materials[6]. Although using lasers reduces this deformation, distortion due to mechanical feed misalignments persists. Changes in the lace pattern are caused also by the release of tension in the lace structure as it is cut. In order to tackle the problem of distortion due to material flexibility in general, a novel approach using *inexact algorithms*, namely *fuzzy logic*, *neural networks* and *neural fuzzy technique*, have been developed. A *Spring Mounted Pen (SMP)* is used in the experiments to emulate the distortion of lace pattern caused by tactile cutting. Using the pre- and post-processing vision stations and the intelligent software kernels, it is possible to monitor the scalloping process as well as generating on-line information fed back to the host system. This allows overcoming the problems of lace distortion due to the trimming operation. Figure 1 shows the data flow diagram for the overview of the pre- and post-processing vision stations. Two A.I. engines are constructed in the system to determine a necessary compensation fed-back to the controller for correcting the errors.

2. CONFIGURATION OF THE SYSTEM

The vision system consists of proprietary components such as two black and white CCD cameras, a four channel video multiplexer, an input/output plug-in card (a video frame grabber, 256K bytes of frame memory organised as 512x512x8 bits) and a video monitor. The above components are integrated within a desktop host environment. A CNC cutting mechanism (Pacer COMPACT 800+) is employed for driving the cutter (or the SMP). In addition an extra conveyor system is fitted on the machine for the transportation of strips of material (lace or paper) under the vision system and the cutter. A M68000 based micro-controller controls both the cutting and the transportation mechanism. The host system bus is used to provide the communication channel among the various elements of the system.

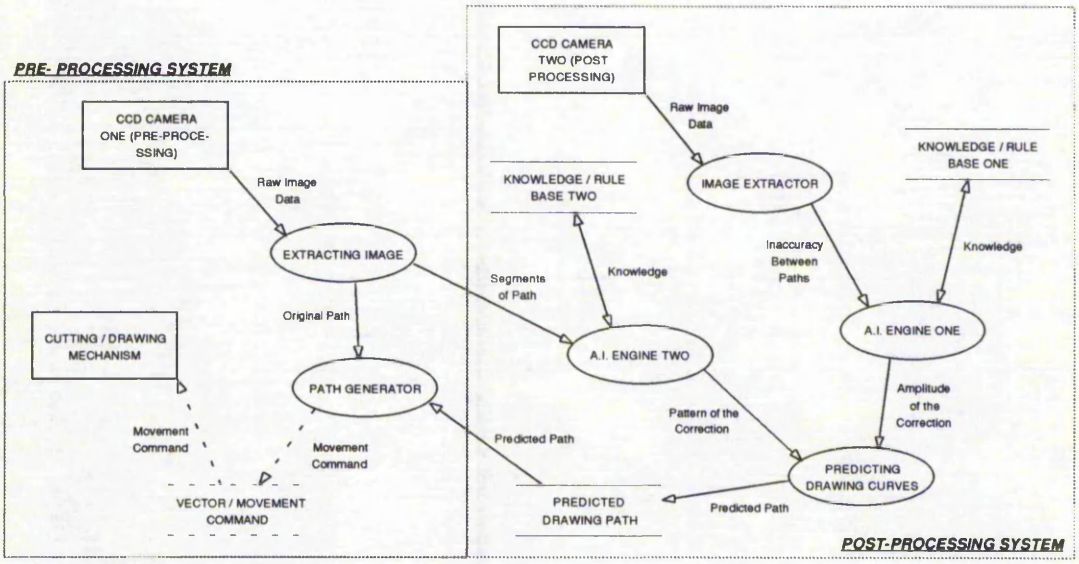


Figure 1: *The overview of the vision and motion control systems*

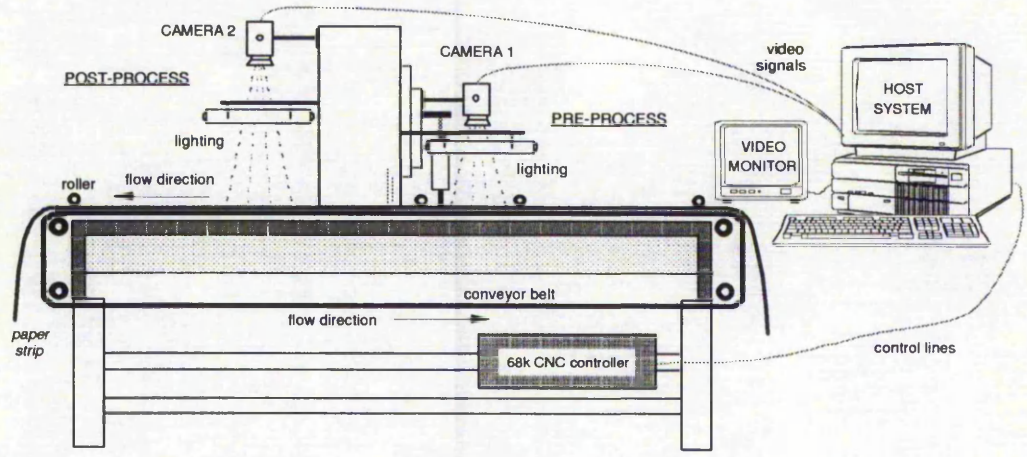


Figure 2: *Schematic diagram of the vision based control station*

Figure 2 illustrates the configuration of the vision based machine control system developed. The host system receives the external video signal which is exchanged by the video multiplexer between two cameras, as well as displaying the captured image on the video monitor. The control data is then generated and transited to the cutting mechanism and the transportation system (conveyor).

A spring is used to mount a pen onto the Z axis of the cutting mechanism (see Figure 3). This device is employed to emulate the distortion of the lace strip. A black line is drawn on paper to emulate the river path within a lace strip. The *pre-processing vision system* captures an image of the paper strip and stores it in the memory. A pattern (intended drawing path) is extracted from the image and transferred to the machine console. This information is used to lead the SMP to draw a second line on the paper strip. As the processed object is passed under the camera of the *post-processing vision system*, an image is taken and analysed to inspect the quality of the SMP following process. The host system takes

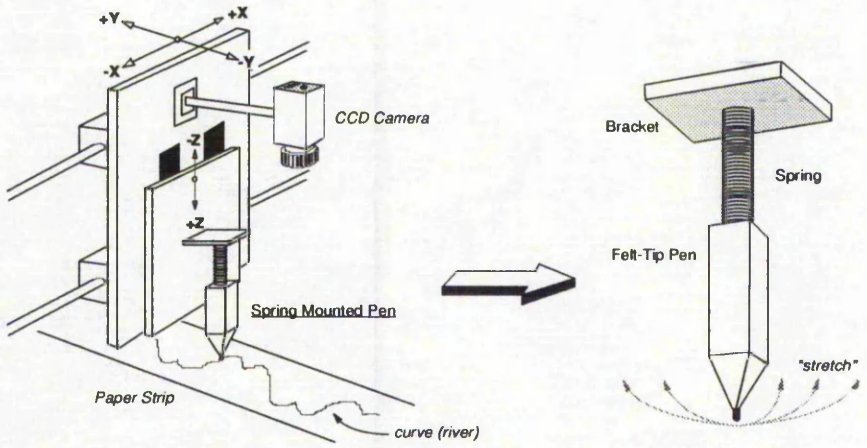


Figure 3: *Spring Mounted Pen (SMP) connected with the testing rig*

this fed-back information, obtained from the post-processing camera, and determines suitable correcting actions to improve this operation.

Figure 4 represents the results of following a square wave using the SMP. Due to the inherent characteristics of the spring, the *path-following-errors* appear between the desired path and the actual target line. Each time the pen is put in contact with paper, the axial load on the spring changes. This consequently causes the pattern generated by the SMP to alter (*path 1* and *path 2* labelled in Figure 4). Path 1 depicts a different Z axis setting to Path 2.

3. PRE-PROCESSING VISION STATION

The pre-processing vision system captures a 256 grey scale image of the target path from *camera one* and temporarily stores it in the memory. An image bi-leveling (thresholding) operation is applied to transform the image into a black and white bitmap. The target path on the paper strip, as indicated in Figure 5, is then extracted by means of a line skeletonisation (skeleton) process. Since conventional camera lenses are employed in the system, the fish-eye lenses create view angle and magnification errors - the captured image tends to be distorted. A software filter is developed and

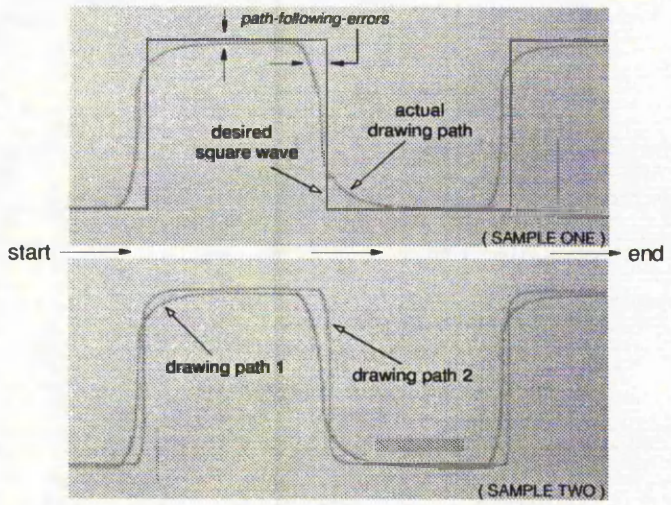


Figure 4: *Samples of square wave following process using the SMP*

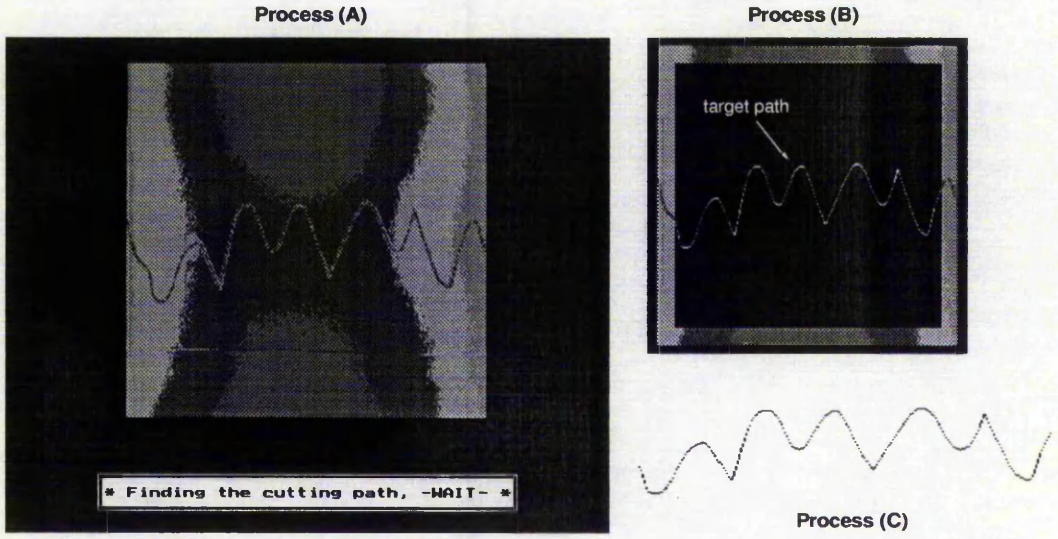


Figure 5: *Processes of the pre-processing vision station; (A) capturing an image, (B) bi-levelling and line skeleton, and (C) correction of distortion and vectorising.*

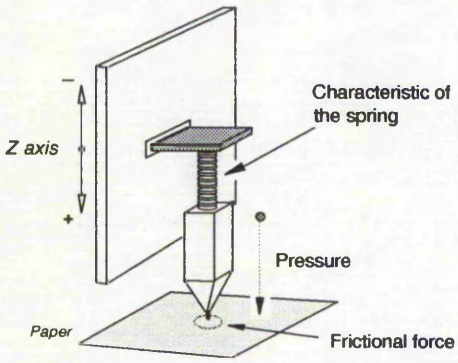


Figure 6: *Three system coefficients affects the SMP following process*

used to correct the distortion caused by the fish-eye lenses. The extracted target path is corrected by the filter and, then, vectorised as well as transferred to the controller of the test rig.

4. POST-PROCESSING VISION STATION

The post-processing vision system is engaged here to capture the image of the generated (processed) path and feed-back the analysed information to the host system. The host PC takes the information to build a statistic record for analysing the operation. This record is then used to improve the process for the subsequent frames (closed loop control).

As the drawn path is transported beneath the camera of the post-processing vision system, an image of the paper strip is captured and stored in the memory. The path-following-error appearing in this frame is detected by a software analyser developed by the authors. Carefully inspecting the results of the SMP following process in Figure 4, it can be seen that the deviation will be generated when the direction of the drawing is changed - the larger the angular variation of the path following, the larger is the error. The amount (magnitude) of the path-following-error generated is depended on the *characteristic of the spring* engaged, the *pressure* on the SMP, and the *frictional force* in between the tip of the pen and the paper (refer to Figure 6). As any one of the system coefficients is altered, the result of the SMP drawing will be entirely different. It seems to be very difficult to use conventional methods, where physical sensors are applied to measure all of the system coefficients hence find the related information to correct this error. In order to overcome the problems of complexity, the inexact algorithms are considered.

5. CLOSELY INTEGRATING THE REMOTE SENSING BASED CONTROL

As stated previously, two cameras together with the CNC machine are integrated using a host computer. The host system supervises the data flows and control actions among all the elements. The tightly integrated system is divided into

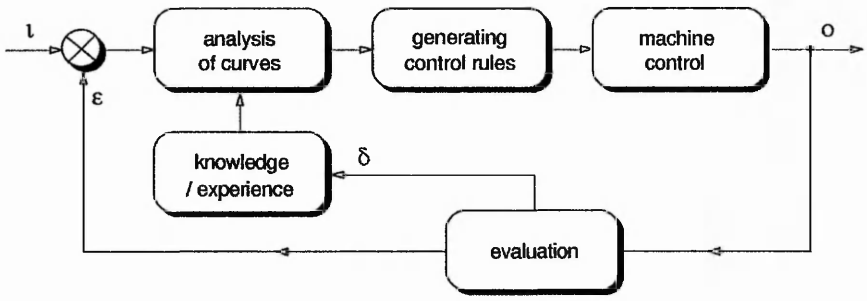


Figure 7: *The control scheme of a close-loop learning process*

three sub-systems: pre-processing vision system, post-processing vision system, and motion control system. In this section, the techniques for incorporating two vision systems to correct the path-following-errors are described.

5.1 The integrated controller

The principle aim is to design an intelligent vision based control system, in the most natural way, in terms of imitating the operator's control action and experience or knowledge. Imagine that the host system controls the cutting mechanism fitted with a SMP to draw a line on paper with no correcting action. Path-following-errors occur between the desired path and the actual drawing line. Utilising the previous experiences, the human operator analyses the difference between the paths and determines a possible correcting action. The operator then controls a joy-stick (X/Y axes) of the machine tracing the desired path as close as possible. This approach, as depicted in Figure 7, can be described as follows:

1) Analysing the difference between two curves, the path-following-error is represented as linguistic description, e.g. *the error between curves on X coordinate is huge*;
2) Expressing the operator's control actions by a set of control rules;
3) Control the machine guiding the SMP to following the desired path;
4) Repeat stages 1) to 3) to get the drawn curve closer to the desired path (learning procedure).

A human operator usually controls a machine based on his experience and/or knowledge which normally can be expressed as a set of control actions (rules). By modeling an operator's control actions and knowledge to design a computer based control system, one does not need to understand how the system parameters (e.g. pressure, frictional force, etc.) physically affect the performance of the controlled system in detail. A driver, for example, who does not need to understand the frictional force between the tires of the car and the road, or the weights of the passengers, can drive the vehicle quite well. Applying this idea to implement the system, the authors proposed an innovative scheme using the inexact algorithms attempting to solve the SMP problems. This approach can be divided into two functional blocks: i) *2VMethod*, and ii) *Piecewise Error Compensation Algorithm*. The fuzzy logic, neural networks and neural fuzzy technique are used to implement this approach, respectively. In this paper we present a summary of the neural fuzzy theory for constructing the A.I. engines.

5.2 2VMethod

As shown in Figure 8, this approach only manipulates a small portion of the intended path to determine a necessary correction for compensating the deviation. By translating a skilled operator's control actions into a set of linguistic terms and groups of network connections, the intelligent machine console can learn from the experiences (on-line learning) and self-adjust the control actions to match the desired objective.

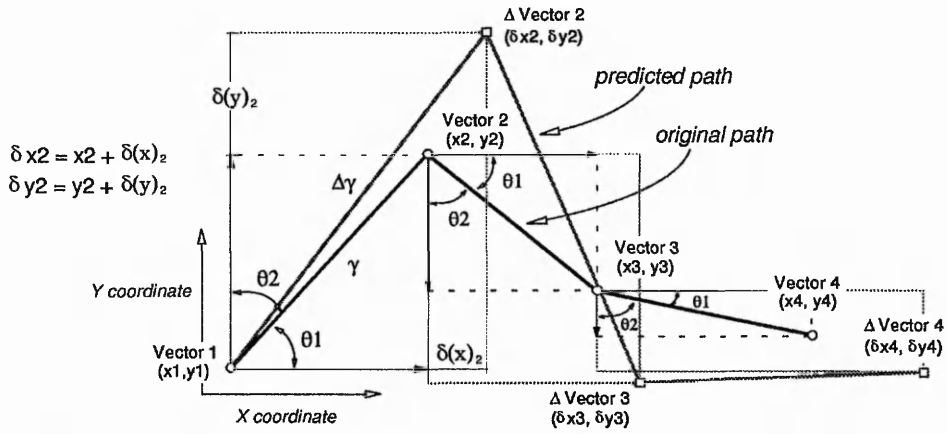


Figure 8: *Calculating new corrected vector from* θ1 *and* θ2 *using the 2VMethod*

Every *two* consecutive vectors are analysed over the entire path. As a straight line is connected from *Vector 1* to *Vector 2* (indicated in Figure 8), the angles (θ1 and θ2) between the line (γ) and the X / Y coordinates are used to compute the possible correcting energies *(δ(x) and δ(y))*. Figure 8 represents the calculation. The angles θ1 and θ2 are related to the each other - θ1 and θ2 are complementary. These two angles are passed to a *neural fuzzy kernel* which is designed by means of applying neural fuzzy theory to determine a suitable compensation (δ(x) and δ(y)). This correction is separated into two "energies" :

IF	Angle is	NL	THEN	correction is	NL
IF	Angle is	NM	THEN	correction is	NM
IF	Angle is	NS	THEN	correction is	NS
IF	Angle is	PS	THEN	correction is	PS
IF	Angle is	PM	THEN	correction is	PM
IF	Angle is	PL	THEN	correction is	PL

Table 1: *Rule base for building the neural fuzzy engines*

i) *Correcting pattern;* and
ii) *Correcting amplitude.*

In the following section, a novel on-line self-learning scheme named the Piecewise Error Compensation Algorithm (PEC Algorithm) based on the neural fuzzy technique derived to calculate the correcting engines is presented.

5.3 Piecewise Error Compensation Algorithm

Two neural fuzzy engines are used by the PEC Algorithm to determine a necessary compensation (correcting pattern and correction amplitude) to eliminate the errors caused by the spring. Table 1 depicts the rules used to build the neural fuzzy engines, where P denotes Positive, N is Negative, L is Large, M is Medium and S is Small. The *neural fuzzy engine one* determines the amounts of amplitudes for the correction. Also, the *neural fuzzy engine two* constructs the correction pattern. Combining these two data sets, the compensated drawing path can be formed.

The neural fuzzy system makes use of neural network for forming the required membership functions and the rule base. As illustrated in Figure 9, the *weight bases* (weights_a and weights_b labelled in Figure 9) of the neural fuzzy system are adjusted and tested by a *neural fuzzy training subsystem.* The system is divided into three functional blocks: obtaining the training data, training the network and testing the trained network. The system updates (tunes) the *weight bases* to form the required system membership functions. After certain iterations of training, the system tests the outputs of the engine. The weights are continuously updated until the outputs of the system match the desired patterns. As the neural fuzzy system is learned successfully, it is employed to form the compensated path.

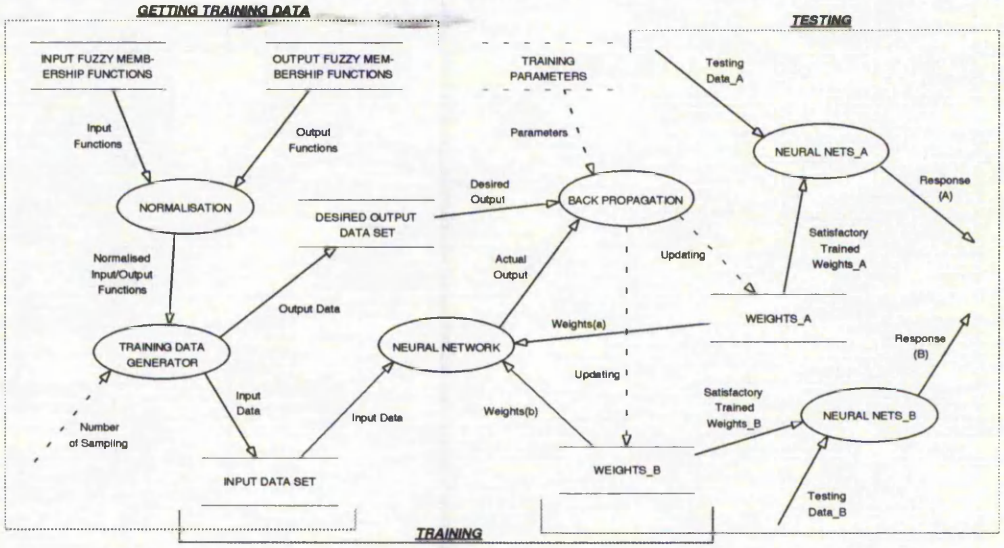


Figure 9: *Neural fuzzy training subsystem*

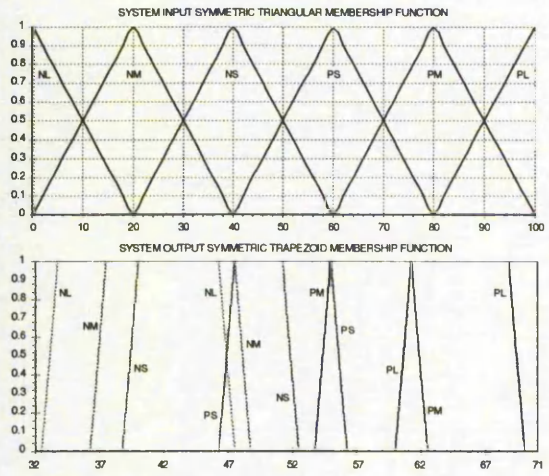


Figure 10: *Membership functions generated by the neural fuzzy engine*

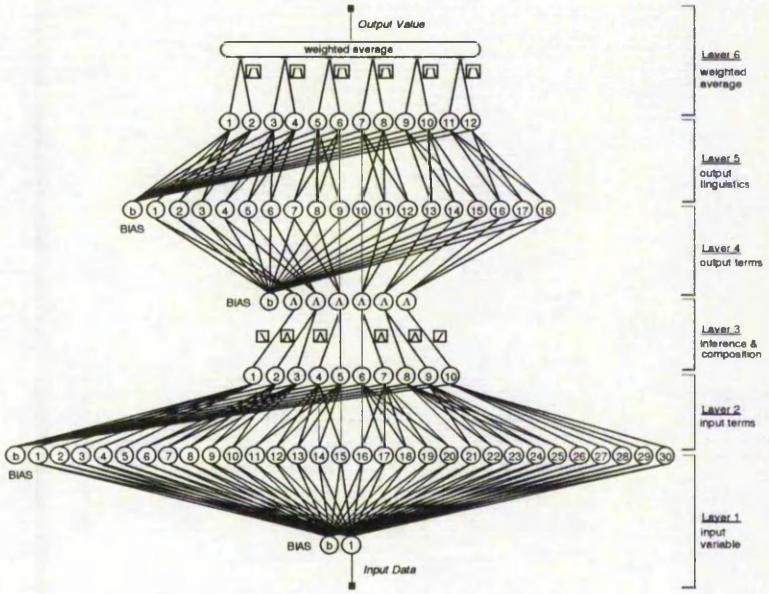


Figure 11: *Architecture of the neural fuzzy kernel*

An example of the system input and output membership functions generated by this novel neural fuzzy architecture is represented in Figure 10. It can be seen that the symmetric triangle and trapezoid membership functions can be successfully created by the network. The combination of these two types of membership functions can enforce the accuracy of the neural fuzzy system responses correctly. By utilising the new network structure better results have been achieved. The results indicate that this scheme performs better than those previously reported[78]. Figure 11 represents the neural fuzzy architecture developed in the project.

The learning process is split into two steps. First, the neural fuzzy kernel one reads the path-following-error from the post-processing vision station and determines a possible correcting amplitude. Then a new detected path from the pre-

processing vision station is fed into the neural fuzzy kernel two to compute the correcting pattern. Equation 1 describes the processes of combining the correcting pattern and amplitude to create a compensated path.

$$PredictedSegment(i) = Pattern_{segment(i)} \cdot Amplitude_{path} \tag{1}$$

where i is the i^{th} segment of the path. The prediction of the new estimated vector $(\Delta Vector2(\delta x2, \delta y2))$ is calculated by Equation 2. Equation 3 presents the procedure for computing the pattern of the predicted path.

$$\delta x(i) = x(i) + \delta(x)_i, \quad \delta y(i) = y(i) + \delta(y)_i \tag{2}$$

$$\begin{aligned} Predicted\ Path &= Vector(1) + \sum_{i=2}^{n} \Delta Vector(i) \\ &= \{x1, y1\} + \sum_{i=2}^{n} \{\delta x(i), \delta y(i)\} \end{aligned} \tag{3}$$

where i is the i^{th} segment of the path and n is the number of vectors in the path.

Once the correction pattern is obtained, the next step is to determine the amount of the amplitude required for the correction. When the first processed frame is passed under the post-processing vision station, an image is taken and sent to the host system. A software recogniser is used to separate the scanned path and the drawing path (see Figure 12). The top, bottom and centre positions in both paths are taken to measure the inaccuracy of the SMP following process. The distances between these points within the different paths are calculated and passed to the neural fuzzy kernel one. The intelligent kernel takes the data and computes the average errors for each of the parameters - *Top, Bottom, SlopeUp,* and *SlopeDown* which are used to determine a suitable amount of amplitude for the correction.

As the neural fuzzy engines are successfully trained, inference engine one reads the path-following-error and determines the amplitude of the correction. Besides, the desired path is fitted into the second neural fuzzy engine which creates the pattern of the correction. Combining the outputs from these two engines (using Equation 1), the compensated path can be formed. Various shapes of cutting paths extracted from the lace patterns captured by the pre-processing vision station have been used to test the developed algorithms. Through the on-line self-learning process, the intelligent host system can determine a necessary correcting path to compensate the deviation.

6. EXPERIMENTAL RESULTS

A working prototype is constructed (Figure 13). Numerous experiments were carried out to evaluate the efficiency of

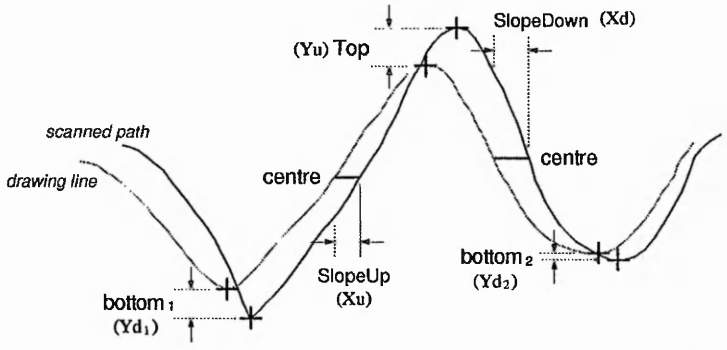


Figure 12: *Detecting the inaccuracy of the SMP following process*

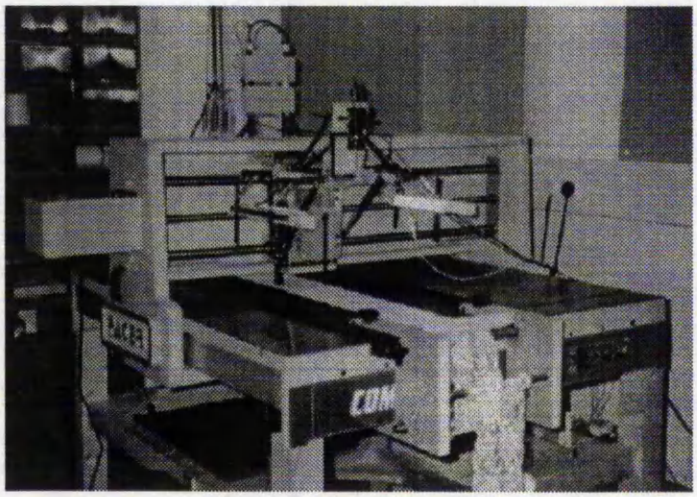


Figure 13: *Prototype of a vision based intelligent control station*

this approach using the 2VMethod and the PEC Algorithm. Irregular shapes of curves are used in the testing. The pre-processing vision station captured an image of the intended drawing path, the host system analyses the image and creates a compensated path for the frame. The post-processing vision station grabs an image of the processed object. The deviation (path-following-error) between the intended path and the resultant path was used to determine a suitable amplitude for the correction.

Figure 14 (A) depicts the drawn line with no correction and its four predicted amplitudes (*New: Yu, Yd, Xu,* and *Xd*) for the correction. These four parameters are used to determine the compensated path for the subsequent frame. Figure 14 (B) shows *Frame One* of the corrected path and its updated amplitudes. The similar process is continuously carried out until the two paths are matched together. Figure 15 (C) and (D) illustrate the Frame Two and Three of the on-line correcting processes. According to various experiments, after three frames of correcting processes almost all the path-following-error caused by the spring can be eliminated. In addition, the obtained (learned) control parameters can be utilised to correct the SMP following errors in the subsequent frames as well as dealing with any regular and irregular shapes of paths. The results indicate that the *2VMethod* applied the *PEC Algorithm* only needs a few frames of self-learning processes to precisely correct for the path-following-errors caused by the SMP, and can produce excellent outcome better than a human operator. Figure 16 illustrates two different regular shapes of paths extracted from the lace patterns and their compensated paths created by the neural fuzzy kernels. To show the algorithms developed can also

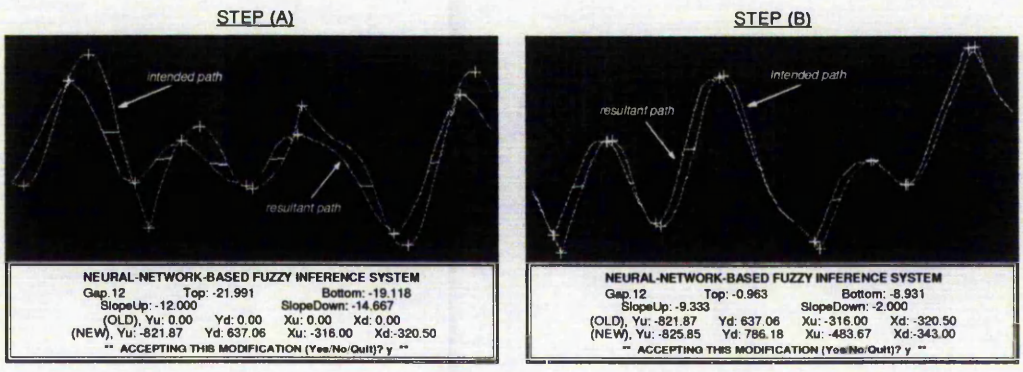


Figure 14: *Using the 2VMethod and the PEC Algorithm to correct the path-following-errors; (A) Frame Zero, (B) Frame One*

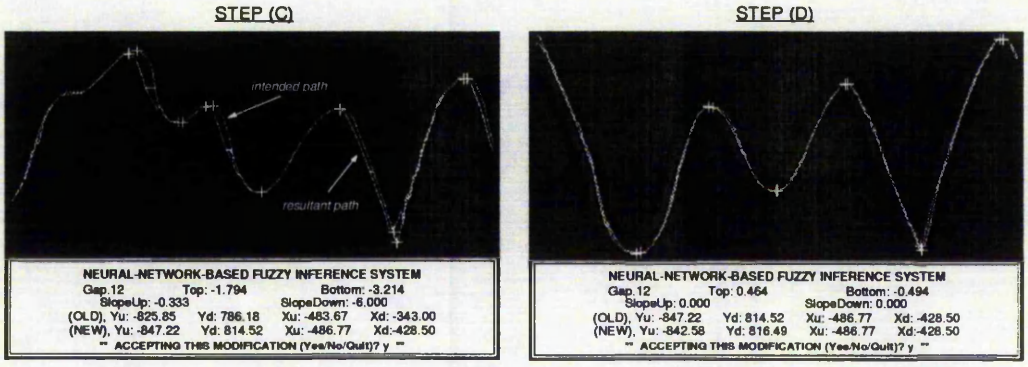


Figure 15: *On-line correcting processes; (C) Frame Two, (D) Frame Three*

handle irregular shapes of patterns, two irregular shapes of paths and their detected compensated patterns are presented in Figure 17.

The development of the intelligent control station is an innovative approach to flexible sheet material processing and has further applications where modeling system behaviour characteristics is difficult. Such systems can range from controlling a robot moving on a slippery surface or piloting a boat. Furthermore, by relying on the intelligent software kernels together with the vision system the controller no longer needs to rely on accurate position feed-back. Backlash, joint flexibility and stick slip can potentially be compensated for by the controller. When characteristics of the mechanism change over time, such as component wear, temperature change, etc., the controller can automatically make appropriate

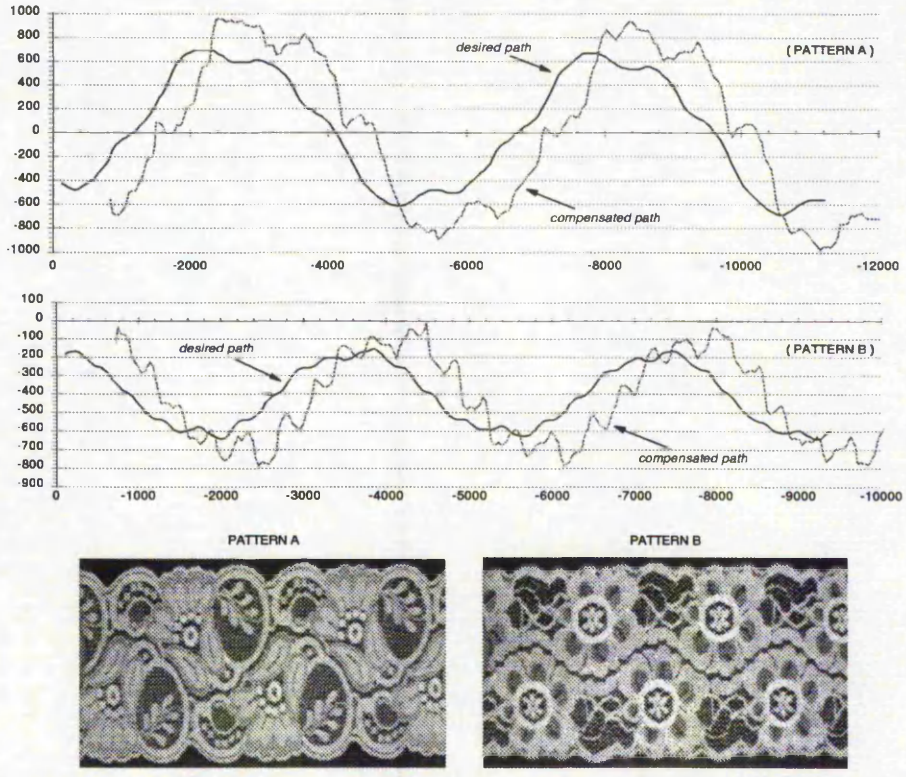


Figure 16: *Two regular shapes of paths extracted from the lace patterns and their compensated paths generated by the neural fuzzy engines*

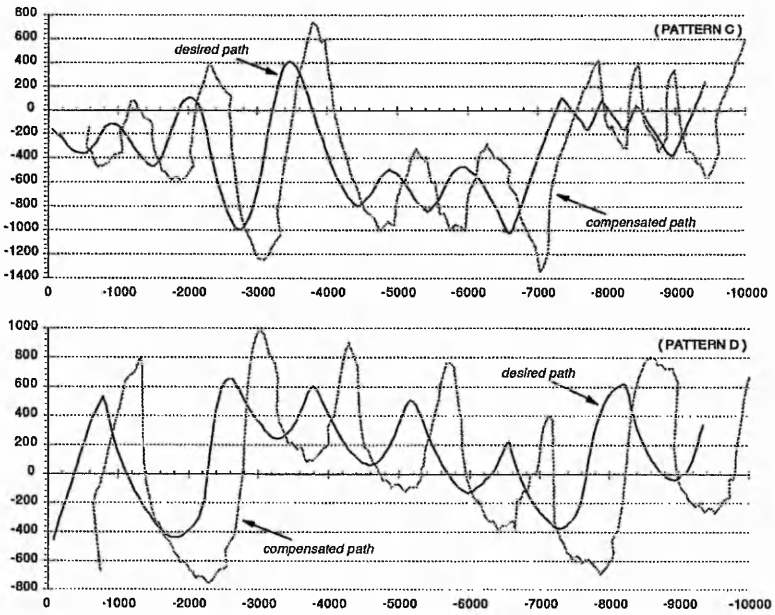


Figure 17: *Two irregular shapes of paths and the detected compensated paths*

compensation.

7. ACKNOWLEDGMENTS

This project has been carried out in collaboration with Axiomatic Technology Ltd. and Pacer Systems Ltd.

8. REFERENCE

1. Sherkat, Nasser; Birch, Mike and Thomas, Peter. "Real-time vision for automatic lace cutting", IS&T/SPIE Symposium on Electronic Imaging Science & Technology, California, USA, 1994.
2. Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology, California, USA, 1995.
3. Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE Fifth International Conference on Image Processing and Its Applications, Edinburgh, U.K., 1995.
4. Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Automation of lace cutting using real-time vision", 8th International Congress on Condition Monitoring and Diagnostic Engineering Management, Queen's University, Canada, 1995.
5. Norton-Wayne. L. "Inspection of lace using machine vision", Computer Graphics Forum, Vol.10, 1991.
6. King, T.; Tao, L.G.; Jackson, M.R. and Preston, M.E. "Real-time tracking of patterns on deformable materials using DSP", IEE International workshop on systems engineering for real-time applications, Royal Agricultural College, Cirencester, U.K, 1993.
7. Chow, M-Y and Goode, P. V. "Adaptation of a neural/fuzzy fault detection system", IEEE Proceedings of the 32nd Conference on Decision and Control, Texas, 1993.
8. Leng, T. S.; Leng, N. S. and Tech, N. W. "Control of an inverted Pendulum using a neuro-fuzzy controller", IEEE: 0-7803-1223-6/93, 1993.
9. Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Tightly coupled vision based control system using inexact algorithms", IEEE Second Asian Conference on Computer Vision, ACCV '95, Singapore, 1995.

Correction of Errors due to Flexibility of Dynamic Systems

Chi-Hsien V. Shih, Nasser Sherkat, and Peter Thomas
Department of Computing, The Nottingham Trent University
Burton Street, Nottingham NG1 4BU, United Kingdom

Abstract

This paper presents an novel approach for tackling problems associated with flexibility of dynamic structures. A number of solutions to this problem have been developed by innovative combination of fuzzy logic and neural networks - inexact algorithms. A Spring Mounted Pen is used in the experiments to emulate the deviation of an end-effector caused by flexibility. A pre- and post-processing vision based machine control system is developed. Comparing the desired pattern and the actual output, the intelligent machine console can automatically make appropriate compensation through on-line self-learning process. Various experimental results indicate that using the algorithms developed the system can compensate for flexibility and produce excellent results, much better than human operators.

1. Introduction

Manufacturing with high accuracy is influenced by numerous factors. These can be classified as follows: machine tool and its controlling equipment, workpiece, fixtures / jigs, tools and environmental conditions. Vibrational errors and control induced errors that appear in a manufacturing system are normally ruled by these factors. Minimising the effect of such errors is usually costly. It would be desirable to rely on the intelligence of the controller to compensate for errors due to flexibility rather than resorting to costly processes of tightening the tolerance.

To tackle the problem of mechanical flexibility in general, a novel approach using *inexact algorithms*, i.e. *fuzzy logic*, *neural networks* and *neural fuzzy technique*, have been developed and described here. A *Spring Mounted Pen (SMP)* is used in the experiments to emulate the movement of an end-effector caused by flexible mechanical structures. Using pre- and post-processing vision systems, it is possible to monitor the processing errors as well as generating on-line information to the Artificial Intelligence engines. This allows overcoming the problems of inaccuracy due to flexibility of dynamic structures. The developed method is essentially trying to avoid using very complex sensors to monitor all the system and other environmental factors, such as mentioned previously. Through a self-learning process - the intelligent kernel compares the difference between the required shape and the resultant shape to make the appropriate compensation.

For example a system which is subject to errors due to flexibility of the workpieces is a lace scalloping machine. A number of attempts have been made to automate the process of lace scalloping and quality inspection [1][2][3]. Work has been reported in using laser technology to cut deformable materials [4]. Although using laser reduces this deformation, distortion due to mechanical feed flexibility and misalignments persists. Changes in the lace pattern are also caused by the release of tension in the lace structure as it is cut. By using the developed algorithm, the problems in lace trimming can be overcome.

2. System Overview

Fig. 1 illustrates the configuration of the developed vision based machine control system. The host system

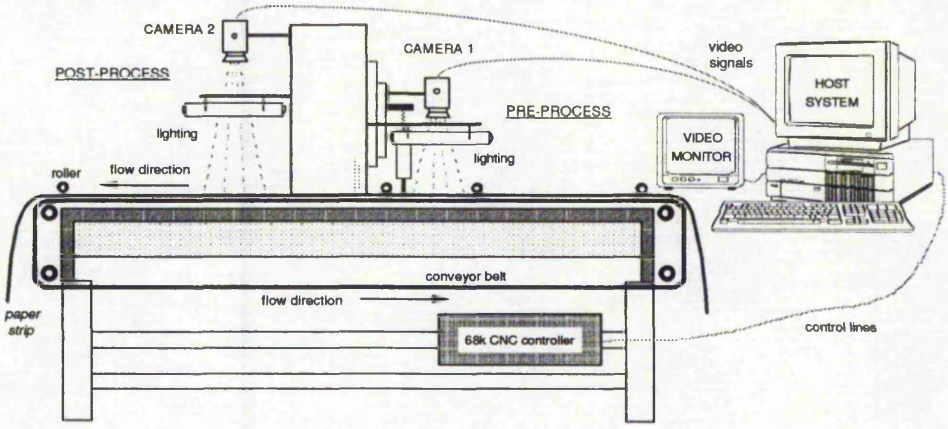


Fig. 1: *Schematic diagram of the test rig*

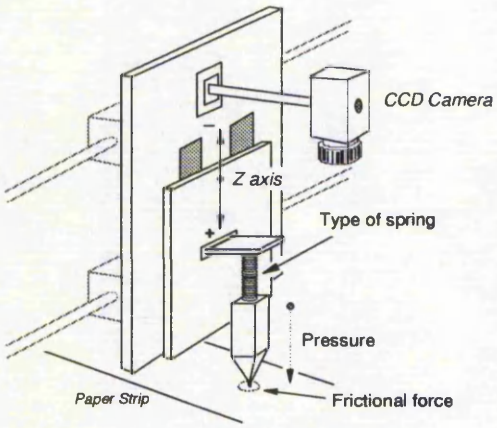


Fig. 2: *Spring Mounted Pen (SMP) connected with the testing rig*

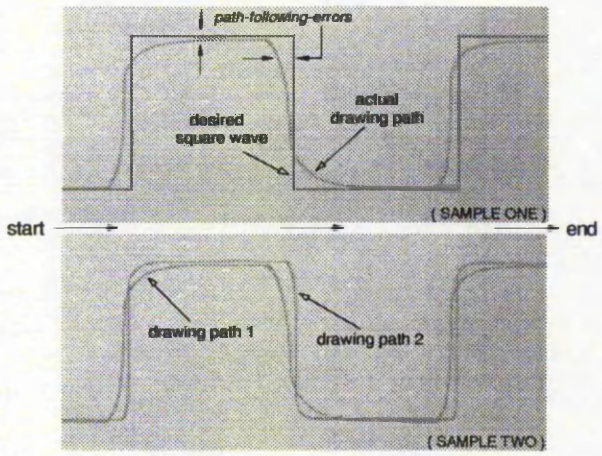


Fig. 3: *Samples of square wave following process*

receives the external video signal which is exchanged by the video multiplexer between the two cameras, as well as displaying the captured image on the video monitor. The control data is then generated and passed to the cutting mechanism and the conveyor system.

A SMP, as depicted in Fig. 2, is guided by the machine to draw a curve on a paper strip to emulate the distortion of the deformable material due to the cutting forces caused by tactile cutting and feed misalignment. Fig. 3 represents the results of following a square wave using the SMP. Due to the inherent characteristics of the spring, the *path-following-errors (PFEs)* appear between the desired path and the actual target line. Each time the pen is put in contact with the paper, the axial load on the spring changes. This consequently causes the pattern generated by the SMP to alter (*path1* and *path2* indicated in Fig. 3).

It is clear that the PFE will be generated when the direction of the drawing is changed - the larger the angular variation of the path following, the larger is the error. The amount (magnitude) of the PFE generated depends upon the *type of the spring* engaged, the *pressure* on the SMP, and the *frictional force* between the head of the pen and the paper (refer to Fig. 2). As any of the system coefficients are altered, the result of the SMP drawing will be different.

3. Integrating the Remote Sensing Based Control

We have proposed a vision based intelligent control system using inexact algorithms. As shown in Fig. 4, the PFE is detected and fed into *A.I. Engine One* which analyses the difference between the paths and decides the *amplitude of correction* for further processing. Camera One (pre-processing system) is triggered to capture a new frame of the desired path on paper. At this time, before the extracted curve is sent to the *path generator*, the segments of the extracted path are passed to *A.I. Engine Two* which determines the correcting *pattern*. Both the *amplitude* and the *pattern* are utilised to generate a *predicted correcting (compensated) path*. Finally, the path generator uses the predicted path and the original path to produce the machine movement data.

According to the information provided by the A.I. Engines, the machine guides the SMP to follow the desired path. A small PFE may still be detected by the post-processing vision system in the first corrected frame. Using the scheme stated above the error information is continuously supplied to the A.I. Engines to calculate more accurate correcting actions until the two paths are finally matched (learning process). An innovative approach named *2VMethod* based on examining each individual segment of the path using inexact algorithms has been developed. These methods form the basis of the compensation system.

4. 2VMethod using Inexact Algorithms

As depicted in Fig. 5, this scheme manipulates a small portion of the detected path to determine a suitable compensation for correcting the PFEs [5]. This method is based on modelling human operators' experience and control actions using the combined fuzzy logic and neural networks. By translating a skilled operator's knowledge into a set of linguistic terms or groups of network connections, the intelligent machine console can learn from the experiences (on-line learning) and self-

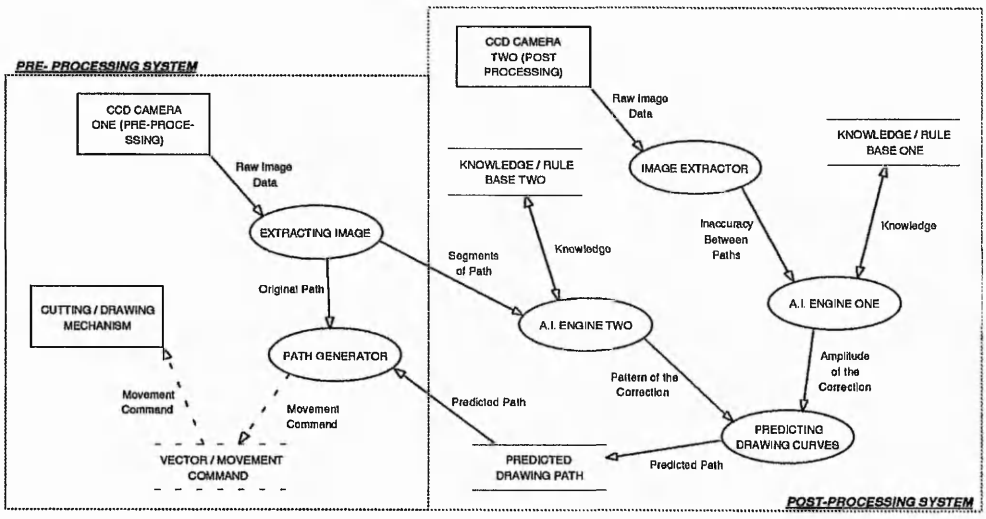


Fig. 4: *The overview of the vision and motion control systems*

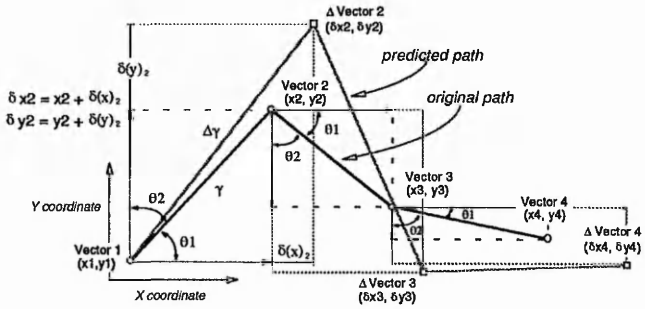


Fig. 5: *Calculating the correcting vectors using the 2VMethod*

adjust the control actions to match the desired objective. Two A.I. engines are constructed in the system to determine a suitable correction. This correction is separated into two "energies":

1) Correcting pattern; and *2) Correcting amplitude.*

Three different approaches, using fuzzy logic, neural networks and neural fuzzy technique respectively, have been applied to determine the correcting energies [6]. In this paper the 2VMethod, employing the neural fuzzy technique to compensate the PFEs, is described.

Fig. 6 shows the scheme using two *Neural Fuzzy Engines (NFEs)* to predict the correcting pattern and amplitude (compensated path). Table 1 lists the rule base used to construct the *NFE*, where N is Negative, P is Positive, L is Large, M is Medium and S is Small. This system is mainly divided into two functional blocks. First, NFE One reads the PFEs from the post-processing vision system and decides a possible amplitude of correction. Then a new detected path from the pre-processing vision system is fed into NFE Two to calculate the correcting pattern. Equation 1 represents the process of combining these two data to create the compensated path.

$$\text{Predicted Segment}(i) = \text{Pattern}_{\text{segment}(i)} \times \text{Amplitude}_{\text{path}} \quad (1)$$

where i is the i^{th} segment of the path.

4.1 Prediction of correcting pattern

To determine the pattern of correction, every *two consecutive vectors* are analysed over the entire path. As a straight line is connected from *Vector 1* to *Vector 2* (refer to Fig. 5), the angles (θ1 and θ2) between the line (γ) and the X / Y coordinates are used to compute the possible correcting energies *(δ(x) and δ(y))*. Fig. 5 represents the calculation [5][6]. The angles θ1 and θ2 are complementary. These two angles are passed to the A.I. Engine Two which is designed by means of using inexact methods to determine the correcting energies (δ(x) and δ(y)). The prediction of the new estimated vector *(ΔVector2(δx2, δy2))* is calculated by Equation 2. Equation 3 describes the procedure of computing the pattern of the predicted path.

IF	Angle is NL	THEN	Correction is NL
IF	Angle is NM	THEN	Correction is NM
IF	Angle is NS	THEN	Correction is NS
IF	Angle is PS	THEN	Correction is PS
IF	Angle is PM	THEN	Correction is PM
IF	Angle is PL	THEN	Correction is PL

Table 1: *Rule base for the NFE*

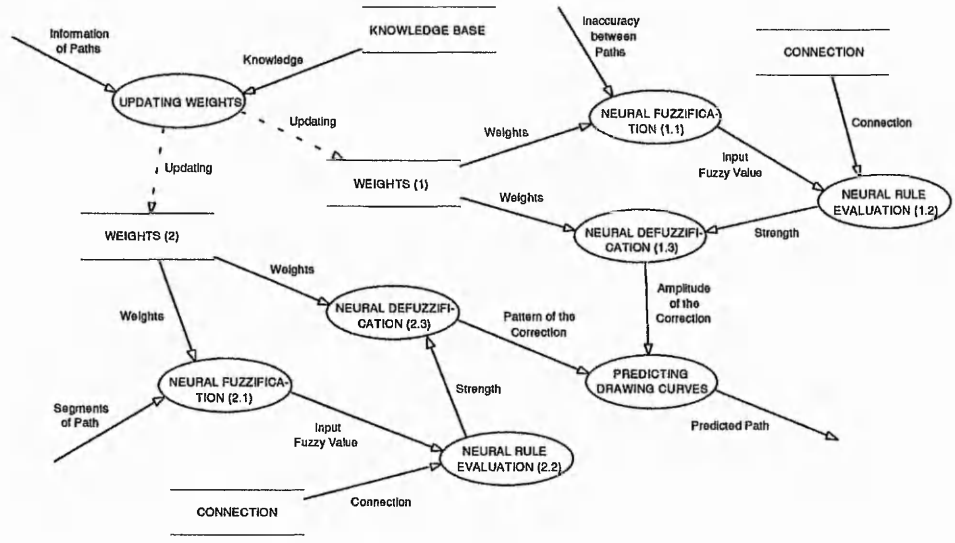


Fig. 6: *Use of neural fuzzy engines to generate a correcting path*

$$\delta x(i) = x(i) + \delta(x)_i, \quad \delta y(i) = y(i) + \delta(y)_i \tag{2}$$

$$\begin{aligned} Predicted\ Path &= Vector\ 1 + \sum_{i=2}^{n} \Delta Vector(i) \\ &= \{x1, y1\} + \sum_{i=2}^{n} \{\delta x(i), \delta y(i)\} \end{aligned} \tag{3}$$

where i is the i^{th} segments of the path and n is the number of vectors in the path.

A tuning process has been developed to adjust the responses of the NFEs into a desired pattern. Fig. 7 depicts an example of tuning the system's responses in terms of analysing the difference between the intended path and the actual path. Carefully inspecting the bell shaped path, two sections of PFEs are observed. The drawn path starts to lose its position at location A (marked by a circle), and re-matches with the desired path at location B. The system calculates the angles of the desired path at the position A and B. After normalising angles Θ1 and Θ2, the regions of the system's response pattern (the darker areas, as indicated in Fig. 7) are modified. By repeating this process, the final forms of the response patterns can be obtained.

4.2 Prediction of correcting amplitude

Once the pattern of correction is determined, the next step is to decide the amount of the amplitude needed for the correction. As the first processed frame is passed under the post-processing vision system, an image is taken and sent to the host system. A software recogniser is employed to distinguish two different colour lines on paper. The top, bottom and centre positions in both paths are taken to measure the inaccuracy of the SMP following (Fig. 8). The distances between these points within the different paths are calculated and passed to the NFE One (see Fig. 6). The engine takes the data and calculates the average errors for each of the parameters -

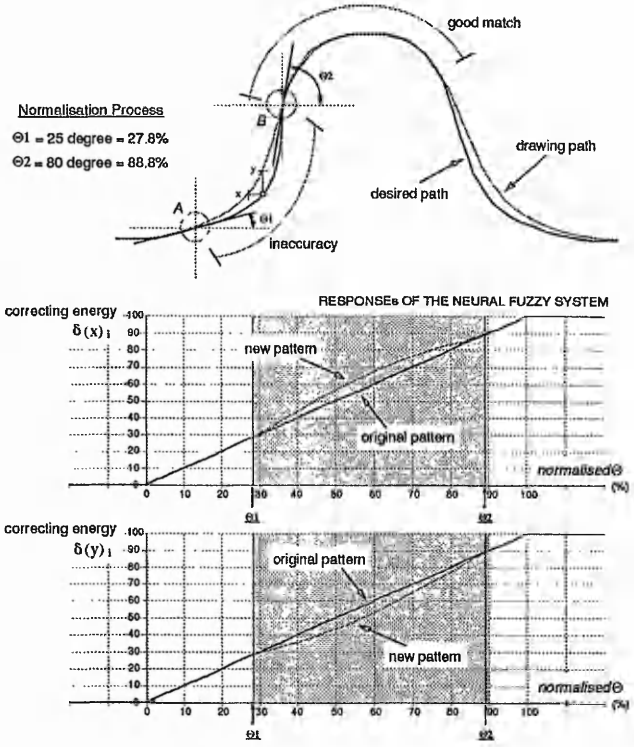


Fig. 7: *Example of tuning the neural fuzzy system's response patterns*

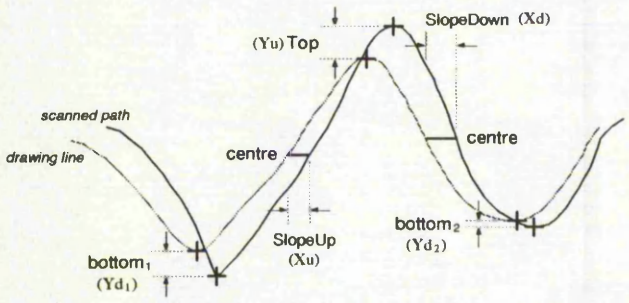


Fig. 8: Detecting the inaccuracy of path following

Top, *Bottom*, *SlopeUp*, and *SlopeDown*. These parameters are then used by the engine to determine a suitable amount of amplitude for the correction.

A novel neural fuzzy architecture, as shown in Fig. 9, has been devised to achieve fast training speed and higher accuracy of response [6]. By applying the new network structure better results have been achieved. The results show that the new method performs better than that previously reported [7][8].

5. Experimental Results

A working prototype is constructed. A number of experiments have been carried out to evaluate the effectiveness of this algorithm. Fig. 10 illustrates two different compensated (correcting) paths generated by the *NFEs* during the learning process. Only less than three frames of learning process are required before the machine reacts correctly to minimise the errors (Fig. 11). Various experimental results indicate that by relying on the algorithms developed the system can deal with any regular and irregular shapes of paths and produce excellent results, much better than a human operator.

6. Conclusion

We have described attempts to develop a vision based intelligent control system for compensating errors due to flexibility of dynamic structures. The development of the system is a novel approach to material processing and has further applications where modeling system behaviour characteristics is difficult, such as to control a robot moving on a slippery surface, drive a car on snow or pilot a boat, etc. Furthermore, by relying on the intelligent software kernel together with the vision system the controller no longer needs to rely on accurate position feedback from the sensors (encoders) on the mechanism. Backlash, joint flexibility, poor feedback and stick slip can potentially be compensated for by the controller. While the characteristics of the mechanism change over time due to component wear, temperature change, etc., the controller can automatically make appropriate compensation.

7. Acknowledgments

This work has been carried out in collaboration with Axiomatic Technology Ltd. and Pacer Systems Ltd.

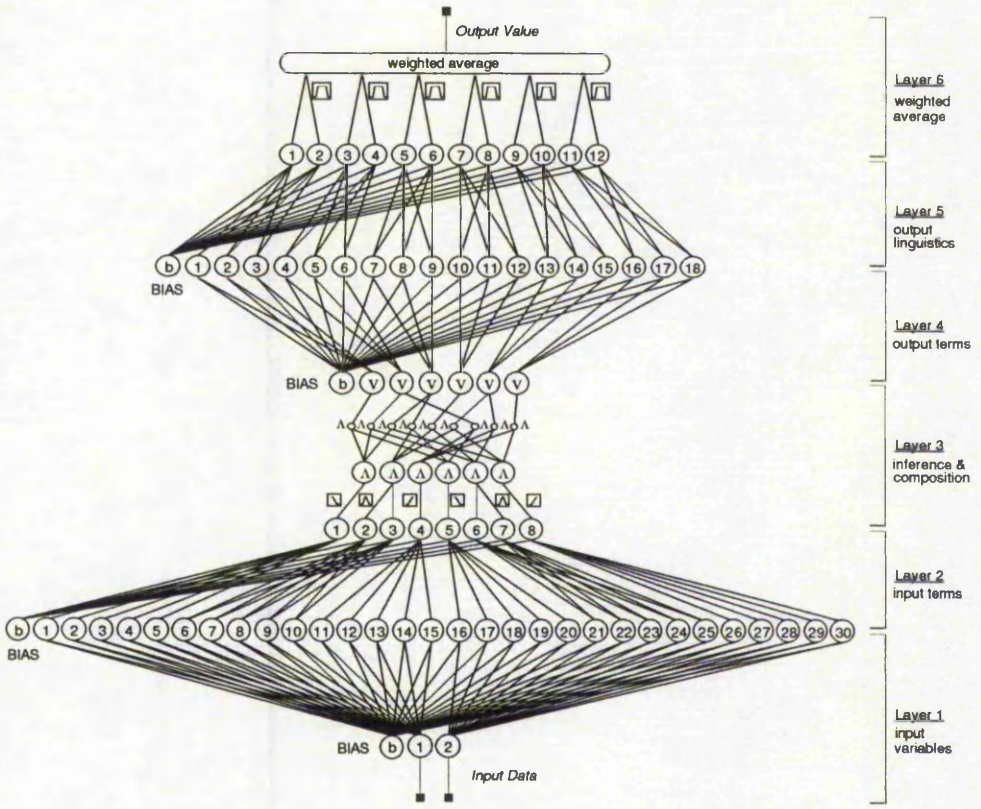


Fig. 9: Architecture of a two-input and one-output neural fuzzy engine

	Amplitudes of Correction			
	Yu	Yd	Xu	Xd
Original Path	0	0	0	0
Compensated Path 1	-435	270	-85	-150
Compensated Path 2	-995	824	-461	-508

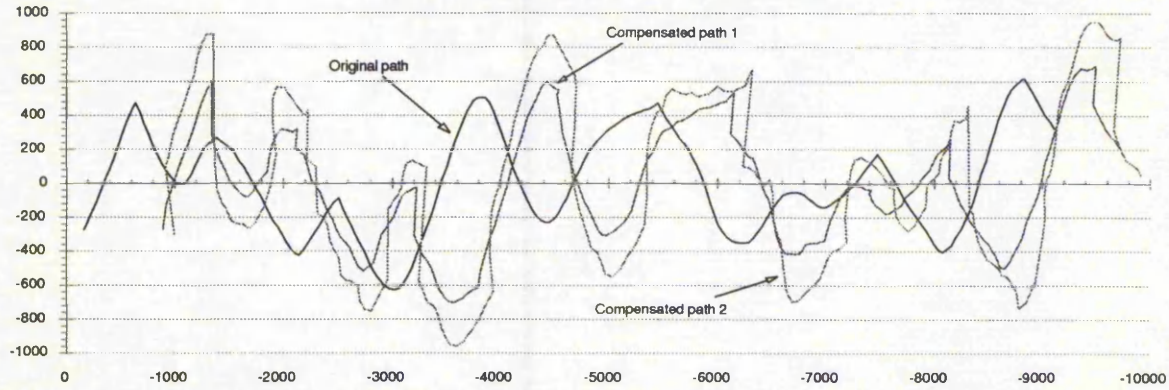


Fig. 10: *Two compensated paths created by the neural fuzzy engines (NFEs)*

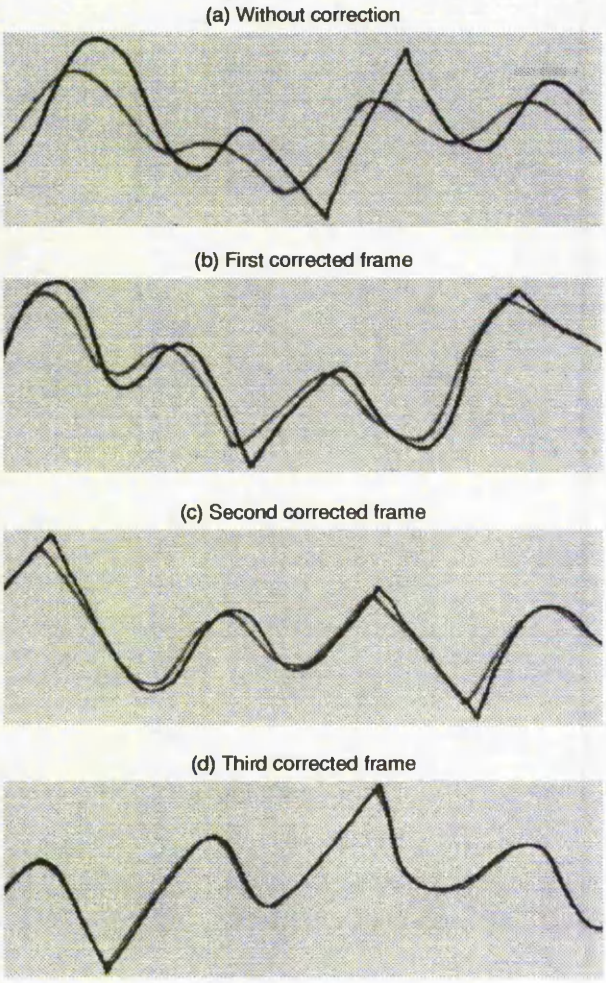


Fig. 11: *Samples of using 2VMethod applied NFEs to correct the PFEs*

References

[1] Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Applications. Kawasaki, Japan, 1994.

[2] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE Fifth International Conference on Image Processing and Its Applications. Edinburgh, U.K., 1995.

[3] Shih, Chi-Hsien V.; Sherkat, N. and Thomas, P. "An automatic lace trimming process using real-time vision". Journal of Real-Time Imaging. 1996.

[4] King, T. "Real-time tracking of patterns on deformable materials using DSP", IEE International workshop on systems engineering for real-time applications. Cirencester, U.K, 1993.

[5] Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "Tightly coupled vision based control system using inexact algorithms", IEEE Second Asian Conference on Computer Vision. Singapore, 1995.

[6] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Close coupling of pre- and post-processing vision stations using Inexact Algorithm", IS&T / SPIE's Symposium on Electronic Imaging : Science & Technology. California, USA, 1996.

[7] Chow, M-Y and Goode, P. V. "Adaptation of a neural/fuzzy fault detection system", IEEE Proceedings of the 32nd Conference on Decision and Control. Texas, 1993.

[8] Leng, T. S.; Leng, N. S. and Tech, N. W. "Control of an inverted Pendulum using a neuro-fuzzy controller", IEEE: 0-7803-1223-6/93, 1993.

AN AUTOMATIC LACE TRIMMING PROCESS USING REAL-TIME VISION

Chi-Hsien Victor Shih*, Nasser Sherkat†, Peter Thomas‡

The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom
Fax.: +44 115 9486518

**Tel.:* +44 115 9418418 ex.4192 **E-mail:* vsh@doc.ntu.ac.uk
†*E-mail:* ns@doc.ntu.ac.uk , ‡*E-mail:* pdt@doc.ntu.ac.uk

ABSTRACT

This paper describe a computer vision based system for automatic lace scalloping. The main problem other than scalloping path detection in real-time is that of coping with material flexibility. This problem varies depending on the material type and the complexity of the lace pattern. The vision system has to work with many different patterns and sizes of lace as well as tolerating misalignment. In order to satisfy industrial requirements two main conditions must be satisfied. To achieve a sufficient degree of automation, first, the river must be found without prior knowledge of the lace pattern being scalloped. A Fuzzy Reasoning Rule-Based Technique is applied to overcome the problems of material distortion. Next, finding the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach called the Line Mapping Method (LMM) is devised to speed up the search for the river in subsequent frames. Several experiments have been carried out using lace patterns of varying complexity. All cutting paths across the patterns were correctly found. Experimental results indicate that the river path can be successfully detected in different lace patterns in real time, while coping with lace distortion.

1. INTRODUCTION

Handling lace in terms of cutting the material along the designed paths is usually carried out manually. Skilled operators use high speed rotating blades or hot wire to cut the lace along the designated path which is illustrated in Figure 1. This is a lengthy and expensive process and results in slowing the rate of production.

Work has been reported in using computer vision for cutting deformable materials. The first published paper was in 1988 [1]. The system employed a low-cost binary vision system which used a Micron Technology IS32 DRAM as its optical sensor to give a 256 by 64 pixel binary image. An Intel 8751 single-chip microcomputer interfaced directly with the IS32 vision sensor. A form of *template matching* was used to determine the position of the lace in the vision system image. The experimental results mentioned in [1] showed that the 8051 microcomputer took about *2 seconds* to perform a template match over the full area of the image. The experimental results indicate that the speed for detecting a cutting path is insufficient for a commercial type of machine. As template matching is used to determine the actual position of the lace image, prior knowledge for each lace pattern is required before the system is operated.

Figure 1: *A typical lace pattern*

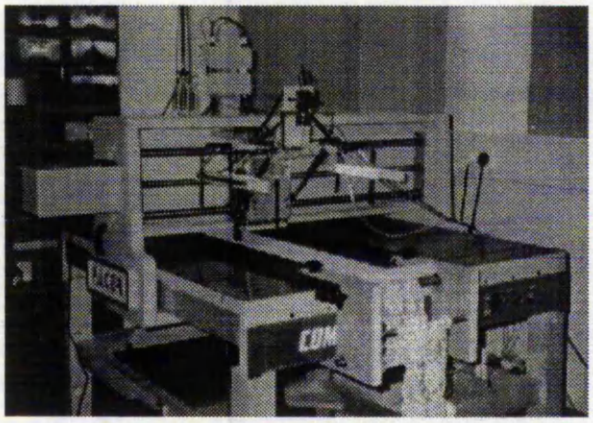


Figure 2: *The prototype of the lace trimming machine*

Another computer vision based lace scalloping system was reported in 1993 [2]. This system was designed for tracking *pre-defined path* along a patterned web of material. The system uses multiple digital signal processors (DSPs) to acquire and process image data from a high-resolution line-scan CCD camera. Control information is generated to allow cutting of the web along the tracked path using a CO_2 laser beam deflected by a galvanometer mounted mirror. Details of the control algorithms used in the system are not reported. According to the reference [2], "Centre-weighted line match (incremental algorithm) and decaying impulse response of a particular filter are used to perform the matching and tracking processes of the lace pattern. Scale errors of ±10% in both directions across the web is allowed by using this approach." Scalloping speeds of 220 mm/s (13.2 meter/minute) are reported. Similar to the previous case, template data from one repeat of the lace pattern is defined manually as a reference map for tracking the actual cutting path. This is considered a disadvantage from the automation point of view.

In 1994, a automatic lace trimming system using real-time vision was developed by the authors (Figure 2). On-line pattern recognition is performed to detect the cutting path automatically (without pre-defining a template). The cutting path is vectorised and transferred to a trimming mechanism. In order to satisfy industrial requirements two main conditions must be satisfied [3]. To achieve a sufficient degree of automation, first, the river must be found without prior knowledge of the lace pattern scanned. A *Fuzzy Reasoning Rule-based technique* is applied to overcome the problems of material distortion [4][5]. Next, finding of the river location across the lace strip must be carried out in real-time. To achieve this, a novel approach named the *Line Mapping Method (LMM)* is used to speed up the search for the river in subsequent frames [6][7].

A bi-level image, shown in Figure 3, is used [3]. After a thresholding operation a river shows up as a dark area (pixel group) within the edges that cross from one side of the image to the other in a nearly unbroken sequence. There are *thick threads* that cross the river at intervals. These are indistinguishable from the material surrounding the river (marked by circles in Figure 3). Allowance must be made for small breaks in continuity of the river due to these cross threads.

Unlike traditional, rigid engineering materials, lace has essentially no stiffness and can shrink, stretch and

Figure 3: *Bi-level lace bitmap image*

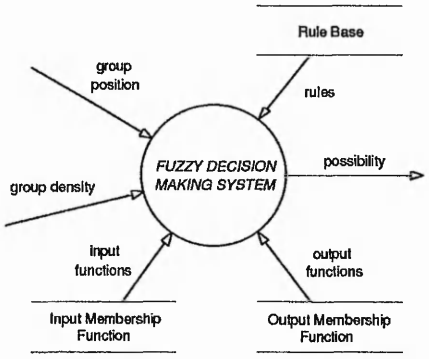


Figure 4: *Context diagram for system overview*

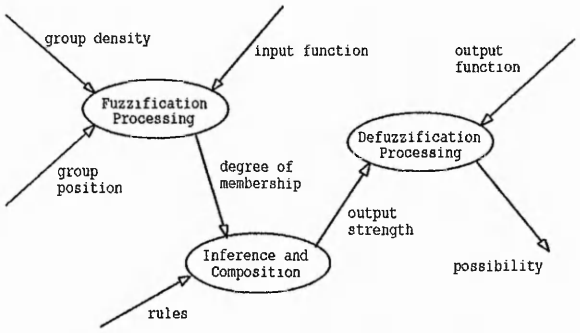


Figure 5: *Second level DFD for decision making process*

distort. To overcome the flexibility problem, we employ an inexact decision making method based on fuzzy rule-based inference technique. As the first cutting river has been recorded, the LMM is engaged to achieve fast detection and higher reliability. The entire system can be broken down into three functional blocks: fuzzy pattern recognition, line mapping process and supervision of the system.

2. Fuzzy Pattern Recognition

The scheme for applying fuzzy inference techniques to find the first river across the lace pattern with no previous knowledge can be broken down into the following tasks:

- Defining system input and output membership functions;
- Fuzzification process;
- Inference and composition;
- Defuzzification process;
- Verification.

Figure 4 illustrates the context data flow diagram of the system. This system reads two input variables *(Group Position and Density)* after each black pixel group has been processed. The fuzzification process then assigns a value to represent an input's degree of membership in one or more fuzzy sets. During inference and composition process, strengths are computed based on antecedent values and then assigned to the rules' fuzzy output. Finally, the defuzzification process employs compromising techniques to calculate the average weight for system output (Figure 5). These steps are described in detail as follows.

1) Defining system input and output membership functions

The degree of membership is decided from overlapping sets of a membership function, which is normally defined based on intuition or experience. The pre-defined membership functions cover the entire range of values for system input and output, and will define a degree of truth for every point in the universe of discourse. As the system is tuned to accomplish desired responses to given inputs or output, it is accepted that membership functions change several times. Nevertheless, once the system is in operation, these membership functions will not be modified. The shapes and number of fuzzy-set membership functions chosen depend on parameters such as the required exactitude, steadiness and responsiveness of the system [8][9]. Different shapes such as triangles and trapezoids are often employed to define fuzzy-set membership functions. Symmetric triangular fuzzy membership functions are applied in the project.

The objective here is to find the river along a lace pattern, by using linguistic variables to represent the common feature of the river shape in various lace patterns. These common features may be described as:

i) the *position* of the river is around the *centre* of the pattern;
ii) the *density* of the river pixel group is not *high*.

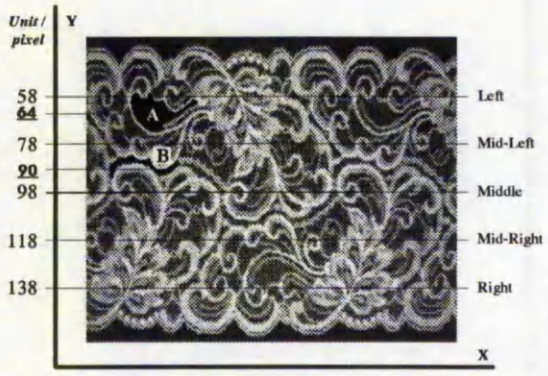


***Figure 6**: Corresponding positions for black pixel group A and B*

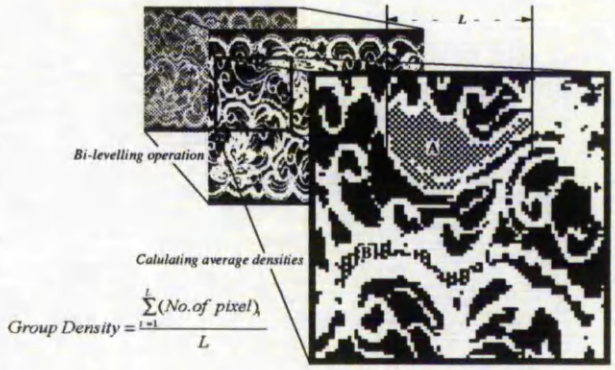


***Figure 7:** A example for calculating the group densities for group A and B*

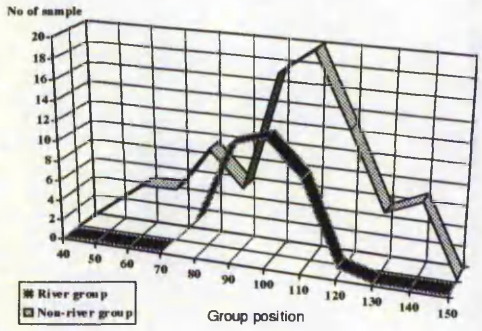


***Figure 8:** Frequency histogram for the position of pixel groups*

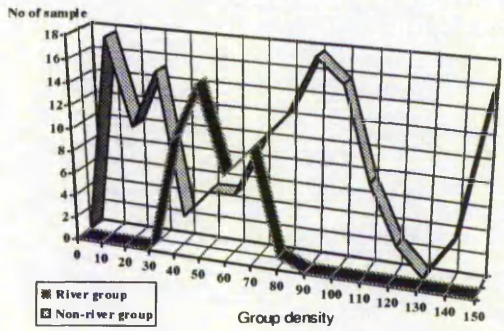


***Figure 9:** Frequency histogram for the densities of pixel groups*

From these linguistic descriptions, two system inputs, *group position* and *group density*, can be defined. By monitoring the position and density of the black pixel groups (Figure 3) across a lace pattern, a fuzzy decision making system can determine whether the pixel group is a possible segment of the river. Figure 6 and Figure 7 illustrate the two system inputs corresponding to an example lace pattern together with two candidate groups A and B.

Two initial experiments were carried out to define the system input and output membership functions. Figure 8 and Figure 9 illustrate the frequency histograms which were taken from the experiments for defining input membership functions [10][11].

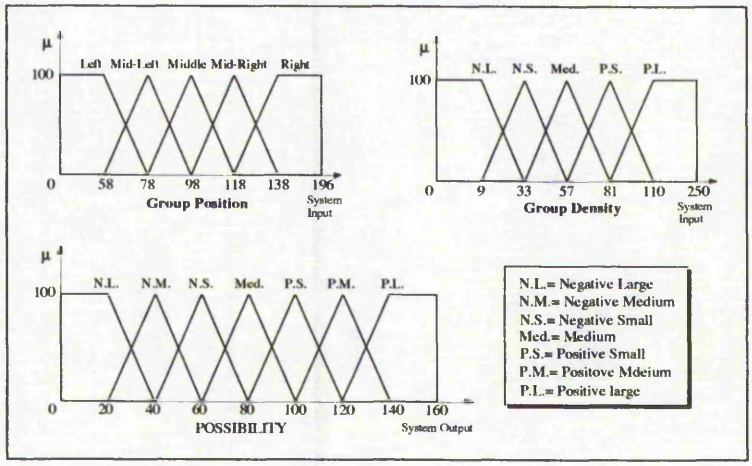


***Figure 10:** System input and output membership functions*

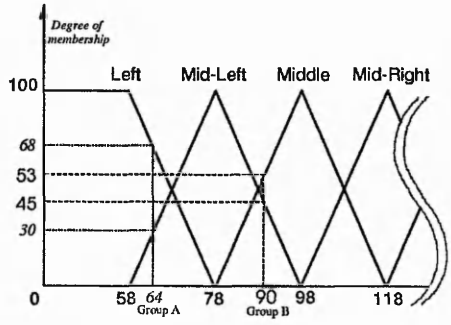


Figure 11: *Fuzzy sets for "group position"*

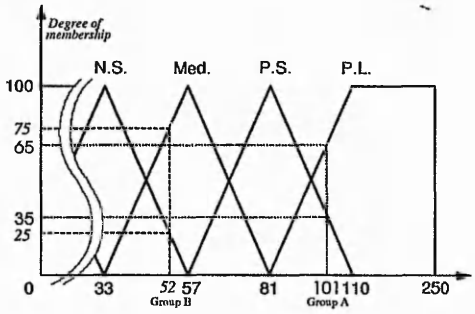


Figure 12: *Fuzzy sets for "group density"*

From these experimental results we can obtain a set of data from the *River group* part (see Figure 8 and 9) to define the membership functions. Triangular membership function is most common and has proved to be a good compromise between effectiveness and efficiency. Overlapping between fuzzy-set boundaries is desirable and the key to smooth operation of the system. To simplify the procedure of defining fuzzy membership functions, an overlap of 50 percent between adjacent fuzzy sets is used in this experiment. In addition to each fuzzy set the central value and the slopes on either side are chosen. Figure 10 shows the fuzzy sets associated with the inputs and output of the system.

2) Fuzzification process

Fuzzification is the procedure of calculating an input value to represent a degree of membership in one or more fuzzy sets. This process uses two basic steps which are repeated for each system input. First, a crisp input has to be read and scaled to a value between 0 and 100. Second, the input must be translated to a degree of membership function. Figures 11 and 12 show two system inputs, *position* and *density*. Each value of system input has a degree of membership in each of these sets. Once the degrees of memberships are assigned, we can utilise these values to evaluate the rules.

3) Inference and composition

Fuzzified inputs are processed through a pre-defined set of rules using min-max evaluation to form fuzzified outputs. The author developed a set of rules that have the form of

Rule 1:	IF position is Left	AND	density is N.L.	THEN	possibility is N.M.
Rule 2:	IF position is Left	AND	density is N.S.	THEN	possibility is P.S.
Rule 3:	IF position is Left	AND	density is Med.	THEN	possibility is P.S.
Rule 4:	IF position is Left	AND	density is P.S.	THEN	possibility is N.M.
Rule 5:	IF position is Left	AND	density is P.L.	THEN	possibility is N.L.
Rule 6:	IF position is Mid-Left	AND	density is N.L.	THEN	possibility is P.S.
Rule 7:	IF position is Mid-Left	AND	density is N.S.	THEN	possibility is P.M.
Rule 8:	IF position is Mid-Left	AND	density is Med.	THEN	possibility is P.L.
Rule 9:	IF position is Mid-Left	AND	density is P.S.	THEN	possibility is Med.
Rule 10:	IF position is Mid-Left	AND	density is P.L.	THEN	possibility is N.L.
Rule 11:	IF position is Middle	AND	density is N.L.	THEN	possibility is P.S.
Rule 12:	IF position is Middle	AND	density is N.S.	THEN	possibility is P.M.
Rule 13:	IF position is Middle	AND	density is Med.	THEN	possibility is P.L.
Rule 14:	IF position is Middle	AND	density is P.S.	THEN	possibility is P.S.
Rule 15:	IF position is Middle	AND	density is P.L.	THEN	possibility is N.L.
Rule 16:	IF position is Mid-Right	AND	density is N.L.	THEN	possibility is P.S.
Rule 17:	IF position is Mid-Right	AND	density is N.S.	THEN	possibility is P.M.
Rule 18:	IF position is Mid-Right	AND	density is Med.	THEN	possibility is P.L.
Rule 19:	IF position is Mid-Right	AND	density is P.S.	THEN	possibility is Med.
Rule 20:	IF position is Mid-Right	AND	density is P.L.	THEN	possibility is N.L.
Rule 21:	IF position is Right	AND	density is N.L.	THEN	possibility is N.M.
Rule 22:	IF position is Right	AND	density is N.S.	THEN	possibility is P.S.
Rule 23:	IF position is Right	AND	density is Med.	THEN	possibility is P.S.
Rule 24:	IF position is Right	AND	density is P.S.	THEN	possibility is N.M.
Rule 25:	IF position is Right	AND	density is P.L.	THEN	possibility is N.L.

Figure 13: *System rule base*

Rule1:	IF position is 68	AND	density is 0	THEN possibility is N.M.
Rule2:	IF position is 68	AND	density is 0	THEN possibility is P.S.
Rule3:	IF position is 68	AND	density is 0	THEN possibility is P.S.
Rule4:	IF position is 68	AND	density is 35	THEN possibility is N.M.
Rule5:	IF position is 68	AND	density is 65	THEN possibility is N.L.
Rule6:	IF position is 30	AND	density is 0	THEN possibility is P.S.
Rule7:	IF position is 30	AND	density is 0	THEN possibility is P.M.
Rule8:	IF position is 30	AND	density is 0	THEN possibility is P.L.
Rule9:	IF position is 30	AND	density is 35	THEN possibility is Med.
Rule10:	IF position is 30	AND	density is 65	THEN possibility is N.L.
Rule11:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule12:	IF position is 0	AND	density is 0	THEN possibility is P.M.
Rule13:	IF position is 0	AND	density is 0	THEN possibility is P.L.
Rule14:	IF position is 0	AND	density is 35	THEN possibility is P.S.
Rule15:	IF position is 0	AND	density is 65	THEN possibility is N.L.
Rule16:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule17:	IF position is 0	AND	density is 0	THEN possibility is P.M.
Rule18:	IF position is 0	AND	density is 0	THEN possibility is P.L.
Rule19:	IF position is 0	AND	density is 35	THEN possibility is Med.
Rule20:	IF position is 0	AND	density is 65	THEN possibility is N.L.
Rule21:	IF position is 0	AND	density is 0	THEN possibility is N.M.
Rule22:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule23:	IF position is 0	AND	density is 0	THEN possibility is P.S.
Rule24:	IF position is 0	AND	density is 35	THEN possibility is N.M.
Rule25:	IF position is 0	AND	density is 65	THEN possibility is N.L.

Figure 14: *Inference and composition for pixel group A*

IF [*antecedent_one*] AND [*antecedent_two*] THEN [*consequence*]

which are listed in Figure 13. The antecedents of rules correspond directly to degrees of membership calculated during the fuzzification process. Each antecedent has a degree of truth assigned to it as a result of fuzzification.

In inference and composition processes, strengths are enumerated based on antecedent values and then assigned to the rules' output strengths. Figure 14 illustrates the actual fuzzy outputs calculated during the rule evaluation process for pixel group A. The strength of a rule is assigned the value of the weakest (*minimum*) antecedent. As more than one rule applies to the same specific action, the strongest (*maximum*) value of rules is used :

a) from Rule 4:

$N.M._{rule\ strength}$ = *min (antecedent_one, antecedent_two)*
= *min (68, 35)* = 35

b) Rule 5:

$N.L._{rule\ strength1}$ = min (68, 65) = 65,

from Rule 10 also

$N.L._{rule\ strength2}$ = min (30, 65) = 30,

the maximum rule strength on fuzzy set N.L. is

$N.L._{rule\ strength}$ = *max (65, 30)* = 65

c) Rule 9:

$Med._{rule\ strength}$ = *min (30, 35)* = 30

In order to further improve the speed of this calculation, the Fuzzy Associative Memory (FAM) Bank [12] is applied to reduce the number of the rules. Inspecting the FAM Bank (Figure 15), the following fuzzy system rule can be formulated:

from rule (A) indicated in Figure 15,

IF the *Group Position* is *Right*
AND the *Group Density* is *Positive Small*
THEN the *Possibility* is *Negative Medium*

This FAM Bank is comprised of 5 × 5 rules. We can reduce the 25 rules per FAM Bank to 11 rules per table by compounding the rules in the Bank. For instance, rule (b) indicated in Figure 15 merges three [*antecedent one*]s of the rules to take the form:

Rule (A): IF Position is Right and Density is P.S.
THEN Possibility is Negative Medium
Rule (B): IF Position is Near Mid. and Density is N.S.
THEN Possibility is Positive Medium

Poss-ibility		Position				
		Left	ML	Mid	MR	Right
Density	NL	NM	PS	PS	PS	NM
	NS	PS	PM (B)	PM	PM	PS
	Med	PS	PL	PL	PL	PS
	PS	NM	Med	PS	Med	NM (A)
	PL	NL	NL	NL	NL	NL

Figure 15: *Fuzzy Associative Memory (FAM) Bank to determine the possibility*

from rule (B),

IF the *Group Position* is *Near Middle*
AND the *Group Density* is *Negative Small*
THEN the *Possibility* is *Positive Medium*

4) Defuzzification process

The defuzzification process is to convert its fuzzy outputs into a single raw or crisp output. There are more than 30 valid defuzzification methods. In these experiments, we choose the *"centre-of-gravity method"* which is a common and accurate defuzzification technique for resolving both the vagueness and conflict issues [8]. Figure 16 is used to illustrate the defuzzification of the output using the centre of gravity method:

a) A centroid point on the x axis is found for each output membership function;
b) The membership functions are limited in height by the applied rule strength;
c) The areas of the membership functions are calculated;
d) The defuzzified outputs are derived by weighted averages of the centroid points and the enumerated areas:

$$Weighted\ average = \frac{\sum(shaded\ area \times centroid\ point)}{\sum(shaded\ area)}$$

By relying on the use of fuzzy inference technique, each black pixel group could be calculated and assigned an average weight (*possibility*). For instance, in Figure 7, the output value for group A is 39.37 (*24.61 %*) (refer to Figure 16), also the group B is 134.64 (*84.15 %*). Since the average weight of group A is only 24.61% (less than 50%), this means that the pixel group only has a 24 percent *possibility* of being a segment of the river. It

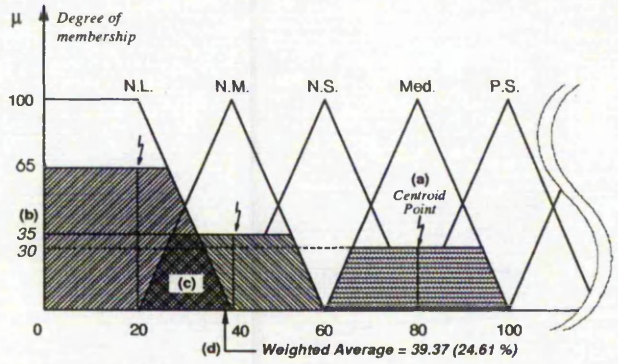


Figure 16: *Defuzzification process for pixel group A*

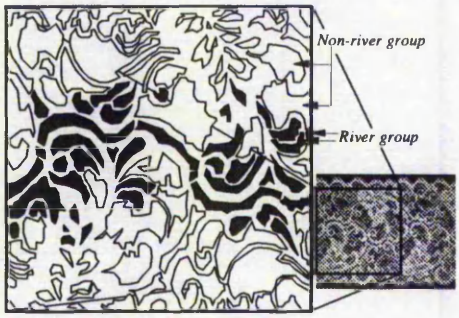


Figure 17: *Each possible river segments whose weights are bigger than 80 (50%)*

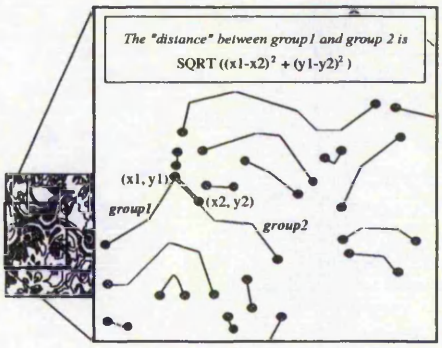


Figure 18 : *An example for calculating the distance between pixel groups*

is, therefore, concluded that group A is not a part of the river.

5) Verification

Once *all* the black pixel groups have been assigned a *possibility value* (average weight), the pixel groups whose possibility values are less than 80 (50%) are abandoned (see Figure 17). The verification process can then be broken down into the following tasks:

a) Calculate the distance between two adjacent groups;
b) If the distance is shorter than a specified value (set to six pixels long in these experiments) a network is built to record this path;
c) Continuously trace the distances between pixel groups while recording all the correct paths until a new pixel group reaches the border of the image (right hand edge of the frame);
d) Calculate the total possibility values and divide by the number of the group in this path (*average possibility*);
e) If the *average possibility* is bigger than a specified value, (110 or 75% was used in the experiments) then the correct river has been found; if the average possibility is less than this value, repeat step (c) to (e) until the correct river is located.

Figure 18 illustrates the computation of the distance between two adjacent pixel groups. By calculating the distances and tracing the average possibilities in all these segments, the river location, highlighted in Figure 19, can be pin-pointed.

3. Line Mapping Process

When the first cutting river in the lace strip is successfully detected, the extracted knowledge can be used to speed up the search in subsequent frames. In order to meet the real-time requirements of the system, instead of using traditional pattern matching techniques, a new approach called the *Line Mapping Method (LMM)* has been developed to achieve fast response and higher reliability. This approach is divided into the following processes:

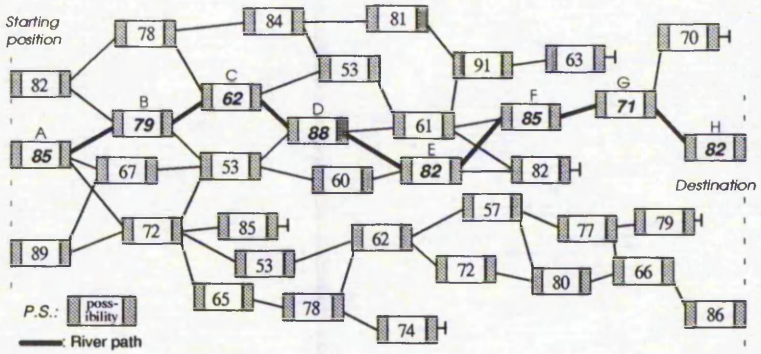


Figure 19: *Interconnection between each possible river segments*

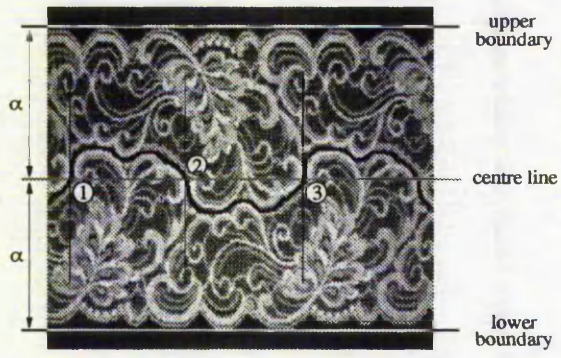


Figure 20: *Extracting a repeat cutting cycle*

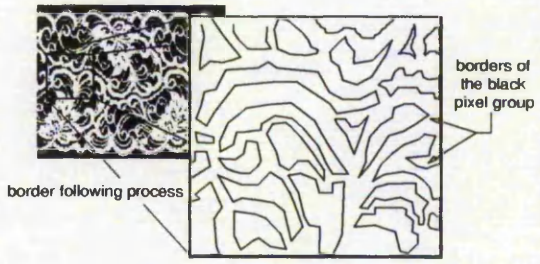


Figure 21: *Borders of the black pixel group*

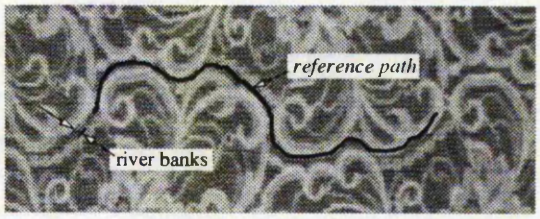


Figure 22: *Mapping the reference path into a new lace image*

1) Indicating and registering one repeat cutting cycle

A *centre line* is located by calculating the distance between the *upper* and *lower boundaries* shown in Figure 20. Three crossing points between the cutting river and the centre line are marked. The cutting path (river) between the intersections ① and ③ labeled in Figure 20 indicates a repeat cutting cycle, which acts as a *reference path* for detection of subsequent frames.

2) Capturing the following frame

The next frame of a 256 grey scale lace image is captured by the frame grabber from the CCD camera and temporarily stored in a memory block. An image thresholding operation is employed to transform the image into a black and white bitmap. This bi-leveled lace image is then applied for detecting the borders of the black pixel groups which may be candidates for river segments. As depicted in Figure 21, the border following technique is used to find the borders (outlines) of the potential river segments.

3) Mapping the reference path into the new frame

Since the lace strip is liable to distort as it is passed through the trimming mechanism, the *reference path* (river) is mapped onto the new frame for the detection of the next cutting river. With careful inspection it is clear, from Figure 22, that the two halves of the image do not completely match (the *reference path* is not

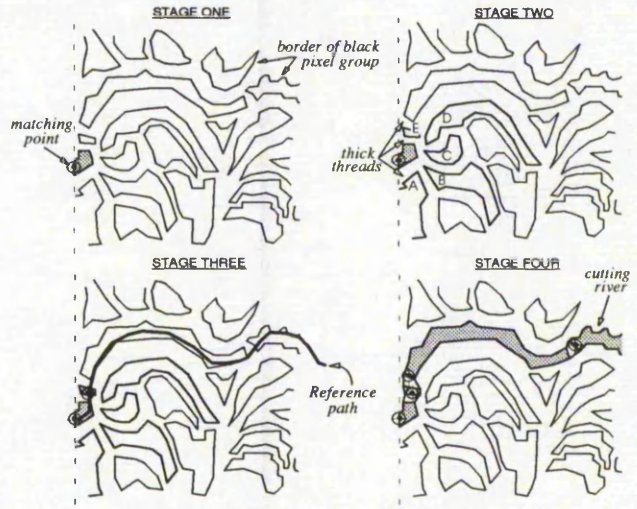


Figure 23: *Using the reference path for searching next cutting path*

completely within the river banks).

A river, as stated previously, crosses from one side of the image to the other in a nearly unbroken sequence. Some allowance has to be made for cross threads produced as a result of the manufacturing process. These are *thick white threads* which cross the river at intervals and are indistinguishable from the material surrounding the river. For this reason, the detection must allow for small breaks in continuity.

The *LMM technique* has been developed for solving this problem. The detection will be started from the left hand side of the frame and ended at the right. As the matching point has been obtained (described in Section 4), the reference path is mapped onto the new frame to find the next border of the river. Several possible connecting borders can be found - A, B, C, D and E labeled in Figure 23 (stage two). The border closest to the mapped reference path is then chosen to become a part of the river (border E is selected in the example). Using the same method, the reference path is repeatedly employed to search the rest of the river segments until it reaches the end of the frame. After all the segments of the river have been found, lines between adjacent river borders are connected, as illustrated in Figure 23 (stage four), the entire river bank can be constructed. By using the detected river bank, a smooth line (the cutting path) can be created within the centre of the river bank.

To summarise the scheme mentioned above:

a) Grab the first frame of the lace image;
b) Use the fuzzy reasoning technique to detect the first cutting river across the pattern;
c) Find the intersections between the centre line and the cutting path;
d) Locate the capture and end of cutting points, respectively;
e) Generate machine movement data from the beginning to the end of cutting position;
f) Start trimming the lace pattern and continuously track the dynamic position of the cutter;
g) Capturing the second lace image when the cutter reaches the capture point;
h) Use fuzzy technique to find the cutting river in the second frame for defining the shape of the *reference path*;
i) Download the machine movement data of the reference path;
j) Using the *LMM* method to map the *reference path* into the third and the subsequent frames of the lace image for fast detection of the repeat cutting path;
k) Continuously trim the strip of the lace into the desired pattern between frames.

4. SUPERVISION OF THE SYSTEM

Since the CCD camera is mounted on the X axis of the machine, the camera is moved with the cutter. The advantage of using such a construction is that the camera and the cutter are kept in a constant position relative to each other. Consequently, it is easy to calculate the real cutting position from the captured image, as well as to correct the errors between these two captured frames. On the other hand, since the lace strip is transported past the vision system by the conveyor belt following the Y axis, the vision system has to consider the more complex two dimensional image shifting problem. Nevertheless, this *"look-and-move"* strategy yields more accurate results than the *"eye-to-hand co-ordination"* approach [13], and also avoids small drift due to material length or

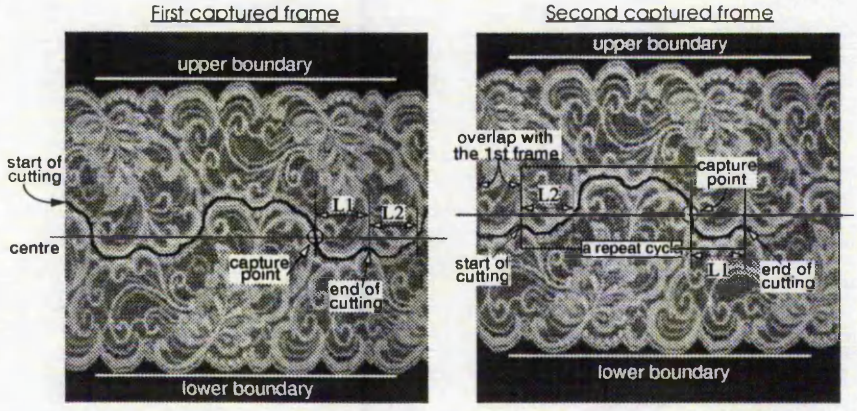


Figure 24: *Vision and cutting procedure*

missing steps of the motor(s). The strategy for analysing images moving in two directions and generating the vector data for machine control is discussed in the following sections.

4.1 Detecting the first river and finding the capture point for next frame

As the lace strip is transported past the field of vision, the first frame of the lace image is captured and temporarily stored in memory. After the fuzzy reasoning process, the cutting rivers across the lace pattern are found. The next stage of the system will then decide the *capture point* on the cutting path for the second lace image. When the machine is in operation, the camera is moving together with the cutter, so finding the position where the camera can capture a similar image for the *LMM* process is critical. As shown in Figure 24, a centre line can be drawn across the first frame, and an intersection between the cutting river can then be found. This position is engaged for grabbing the second frame of the lace image.

4.2 Generating machine movement data and grabbing the second frame

While the first cutting river has been detected, using the fuzzy reasoning method, the machine control data is generated and downloaded to the machine controller. The controller transforms the motion data into the real machine movement data and starts driving the cutter to cut the strip of lace. When the machine starts cutting the lace strip, the controller simultaneously responds to the machine console with the current position on the XY axes. The machine console then continuously tracks the cutting positions until it reaches the *capture point* (shown in Figure 24 - first captured frame). Consequently, the CCD camera is triggered to capture the second frame of the lace image which is stored into memory for processing.

Since the system takes approximately two hundred milli-seconds to find the next cutting river, this will stop the cutting process between two captured frames. To solve this problem, we simply add a quarter of the repeat cutting cycle (*L1*, between *capture* and *end of cutting points*, indicated in Figure 24) to the cutting path. Thus, while the machine is trimming past the *capture point*, the vision system grabs a frame as well as finding the cutting river before the machine actually ends trimming. This enables continuous operation of the system in real-time.

4.3 Finding the reference path and the next capture point

As the second lace image is stored in the memory, the fuzzy reasoning rule-based technique is, again, employed to find the second frame of the lace image. The second intersection with the cutting river can be designated as the *capture point* for the next frame. As the machine continuously trims the lace and reaches the *'end of cutting'* position in the first frame, the movement data of the second cutting path has already been produced and stored. Therefore, the machine could continuously cut the lace pattern through subsequent frames in an unbroken sequence.

L1 and L2 indicated in Figure 24 are taken from the first frame, and coupled to the second frame for determining the length and shape of the reference path (a repeat cutting cycle). After the reference path has been defined, its corresponding position with the centre line (first pixel of this module) is then registered. This will be used for detection of subsequent frames.

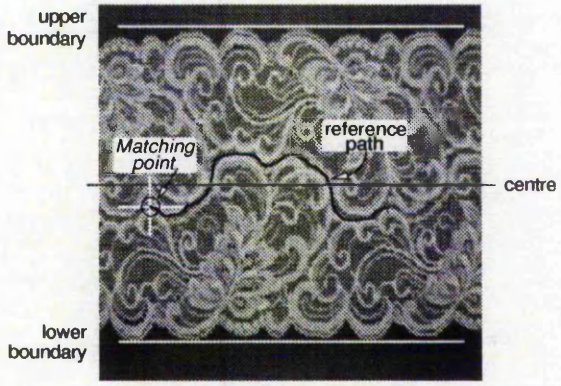


Figure 25: *Mapping the reference path into subsequent frame*

Figure 26: *Example A of river extraction*

Figure 27: *Example B of river extraction*

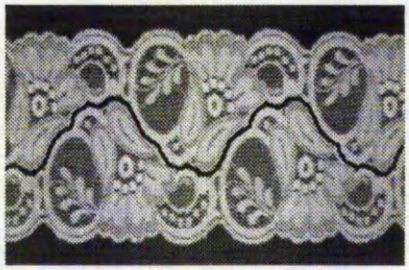


Figure 28: *Example C of river extraction*

4.4 Line mapping operation

After the *reference path* has been determined, the extracted knowledge can be used to speed up the search for the river in subsequent frames. Figure 25 shows an example of mapping the reference path into the following frame. Utilising the *Line Mapping Method (LMM)*, the new cutting river across the lace image can be successfully and quickly detected.

5. Experimental Results

Various experiments were carried out to investigate the effectiveness of this method. Numerous lace patterns were employed for detecting the river location. All cutting paths across the patterns were successfully found. The time taken to isolate the river and produce cutting path depends on complexity of the pattern. Time taken for most kinds of motif, using the fuzzy reasoning rule-based technique, is typically about 300 milli-seconds using an Intel 80486 processor running at 66 MHz. Nevertheless, in the case of a very few intricate lace patterns (e.g. Figure 26), up to 1.5 seconds is required.

Once the river path on the first frame is found, this knowledge can be utilised by the LMM to speed up the detection for the river in subsequent frames in real time. The time to detect a repeat cutting path using LMM is dependent on how complex the motif is, the length of one repeat cutting cycle and the distortion of the pattern. On most kind of lace patterns detection time is about 150 to 200 milli-seconds. The frame grabber digitises a incoming video signal at a rate of 30 frames per second. Typically a repeat cutting cycle of the lace strip is around 9 to 15 cm. Therefore the speed for tracking the lace pattern using the LMM is approximately 25 to 35 meters / minute. Some sample lace patterns together with the resulting river path are shown in Figures 26 to 28.

The strip of lace is likely to stretch or contract while it is passed through the machine via the feed

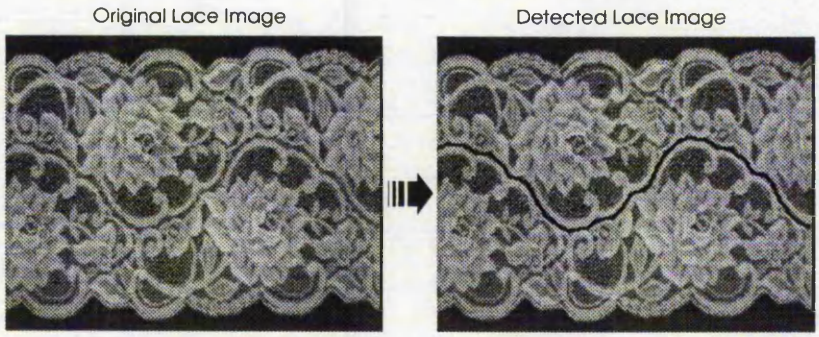


Figure 29: *Un-distorted lace pattern*

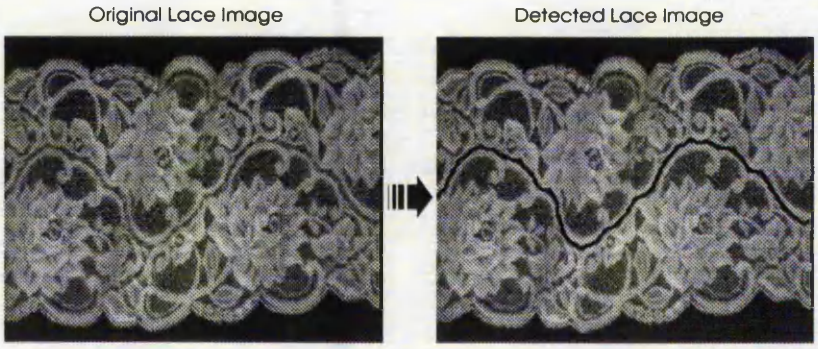


Figure 30: *Lace pattern under 30% contraction (successfully detected)*

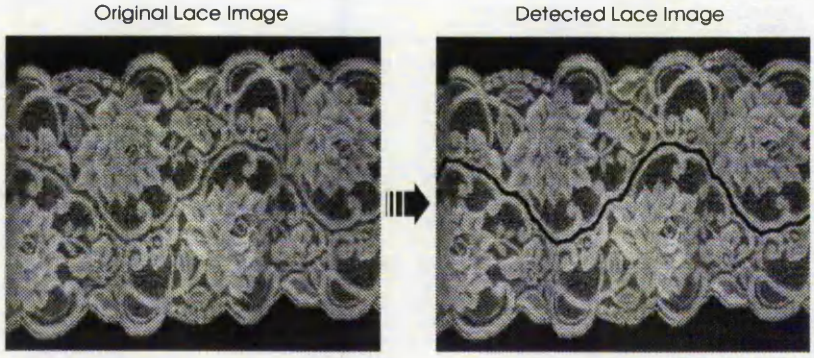


Figure 31: *Lace pattern under 40% contraction (successfully detected)*

mechanism. Several experiments have been carried out for investigating the capability of this approach. Various kinds of lace patterns have been examined under the following status: 1) Non-distortion, 2) contraction, 3) and stretch.

1) Non-distortion

Under this condition, all cutting paths across the lace patterns were successfully found. Figure 29 illustrates a typical lace pattern as well as its detected cutting path.

2) Contraction

Figure 30 and Figure 31 show that the lace motifs have been contracted with 30 per-cent and 40 per-cent of the pattern. The cutting paths within these two frames have been successfully detected by applying the novel techniques. On the other hand, when we use traditional image processing methods [3] to examine these distorted patterns, according to the experimental results, none of them could find the river, or even pinpoint the wrong paths.

The feature of the lace motif under 50 per-cent of contraction is illustrated in Figure 32. With careful inspection of this picture we can find that parts of the river banks are overlapped, it is difficult to find a nearly unbroken river across the strip of the lace. In the real working situation, although the skilled human operator could find the cutting path from the contracted lace pattern, it would be impossible to separate this pattern correctly by using knife or lace cutter. Consequently we can ignore this situation in our experiments.

3) Stretch

Lace was stretched lengthwise in order to emulate stretch resulting from lace transport. Figure 33 shows that the lace pattern has been stretched as much as possible, and its detected river across the pattern. Typically, about 15 to 30 per-cent of lace can be stretched, depending on material and patterns.

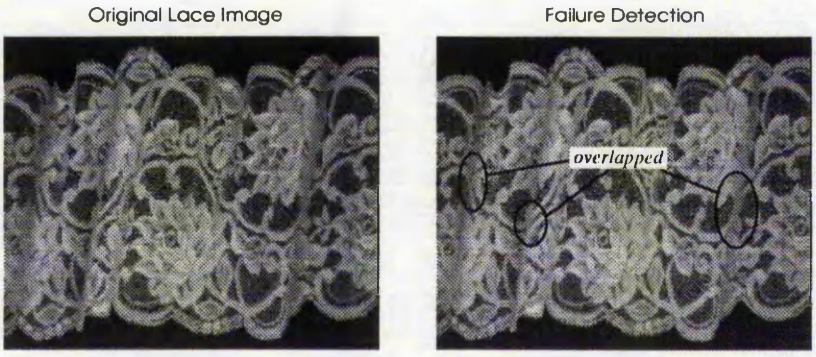


Figure 32: *Lace pattern under more than 50% contraction (fail to detect the correct cutting path)*

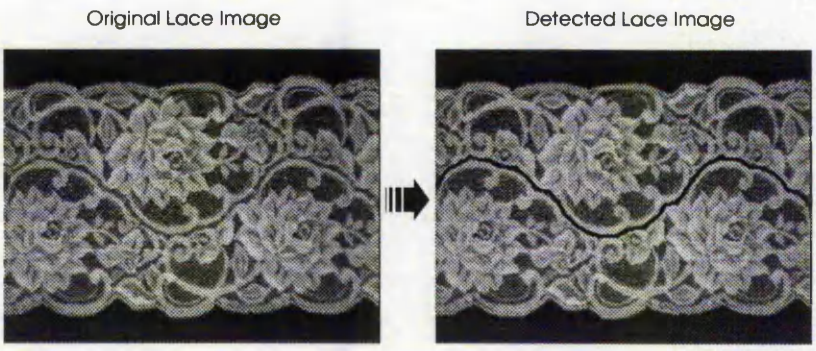


Figure 33: *Lace pattern under maximum stretch*

Compared with the previously reported method [3], the fuzzy logic based approach is more effective. The traditional image processing technique for finding the river heavily depends on the repeat of the lace pattern. In other words, the two extremes of the river should be equi-distant from their nearest edge, and after a distance equal to the repeat period of the design, the river should be back at the same position relative to the two edges as it was at the start. When lace is distorted, these features of the river are absent. That is why the conventional method fails when strips of lace are slightly distorted (5- 10%).

6. Conclusion

In the preceding sections of this paper, we have described attempts to develop a *fuzzy reasoning rule-based system* and the *line mapping method* for detecting varied types of lace patterns in real time. Experimental results indicate that the objectives have mostly been fulfilled. The system requires no prior knowledge of any particular lace pattern or any training. According to the results of the experiments, a combination of fuzzy pattern recognition technique and the LMM can be used to successfully detect the cutting river within various lace patterns in real time. Compared with the conventional image processing methods [3][12], it is not only easier to design and implement the system, but also more effective in coping with distortion. Furthermore it does not require any training or prior knowledge of the lace pattern.

References

[1] Russell, R.A. and Wong, P. "Automation of lace cutting using computer vision", Robots: Coming of Age. Proceedings of International Symposium and Exposition on Robots. Designed the 19th ISIR by the International Federation of Robotics, Nov. 1988, pp.385-93.

[2] King, T.; Tao, L.G.; Jackson, M.R. and Preston, M.E. "Real-time tracking of patterns on deformable materials using DSP", IEE International workshop on systems engineering for real-time applications, Royal Agricultural College, Cirencester, U.K, 1993, pp.178-183.

[3] Sherkat, Nasser; Birch, Mike and Thomas, Peter. "Real-time vision for automatic lace cutting", IS&T/SPIE Symposium on Electronic Imaging Science & Technology, 1994, pp.322-333.

[4] Sherkat, Nasser; Shih, Chi-Hsien Victor and Thomas, Peter. "A fuzzy reasoning rule-based system for lace pattern detection", IAPR Workshop on Machine Vision Application, Kawasaki, Japan, 1994, pp.61-64.

[5] Sherkat, Nasser; Shih, Chi-Hsien Victor and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", IS&T/SPIE's Symposium on Electronic Imaging : Science & Technology, California, USA, 1995, pp.323-333.

[6] Shih, Chi-Hsien Victor; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", IEE's Fifth International Conference on Image Processing and Its Applications, Edinburgh, UK, 1995.

[7] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Automation of lace cutting using real-time vision", 8th International Congress on Condition Monitoring and Diagnostic Engineering Management, Queen's University, Canada, 1995.

[8] Zadeh, Lofti A. "Fuzzy logic", Computer, IEEE, April 1988, pp.83-93.

[9] Zimmermann, H. J. Fuzzy Sets, Decision Making, and Expert Systems, Kluwer Academic Publishers, Lancaster, 1987.

[10] Roberts, D.W. "Analysis of forest succession with fuzzy graph theory", Ecological Modeling, 45:261-274, 1989.

[11] Turksen, I.B. "Measurement of fuzziness: interpretation of the axioms of measure", Proceeding of the Conference on Fuzzy Information and Knowledge Representation for Decision Analysis, IFAC, Oxford, 1984, pp.97-102.

[12] Nedungadi, Ashok and Wenzel, Dennis J. "A novel approach robot control using fuzzy logic", IEEE, ISSN#0-7803-0233-8/91, 1991.

[13] Wolfe, D.F.H.; Wijesoma, S.W. and Richards, R.J. "Eye-to-hand co-ordination", Assembly Automation, Vol.11, No.1, 1991, pp.15-20.

Inexact Algorithms for Correction of Errors due to Flexibility of Dynamic Structures

Chi-Hsien V. Shih*, Nasser Sherkat†, Peter Thomas‡
The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom
Fax.: +44 115 9486518, *Phone: +44 115 9418418 ex.4192
*E-mail: vsh@doc.ntu.ac.uk, †E-mail: ns@doc.ntu.ac.uk , ‡E-mail: pdt@doc.ntu.ac.uk

Abstract **– In the development of an intelligent controller to manage the flexibility of dynamic systems, the combination of fuzzy logic and neural networks, inexact algorithms, can be applied in constructing an intelligent kernel. Using human operator's experience and knowledge, correcting actions can be translated into groups of network connections and sets of control rules. In order to emulate the distortion of an end-effector caused by flexible mechanical structures, a *Spring Mounted Pen (SMP)* is designed and engaged in the experiments. A machine vision station is constructed in the test rig to monitor the results of the SMP following process. By analysing the processing errors as well as feeding back on-line information to the intelligent controller, the problems of inaccuracy due to flexibility of dynamic systems can be overcome. The method developed is essentially trying to avoid using very complex sensors to monitor all the system and environment factors. Through a on-line self-learning process - the intelligent kernel compares the difference between the desired pattern and the resultant pattern to make the appropriate compensation.**

I. Introduction

In high accuracy manufacturing systems, control induced and vibrational errors are normally ruled by various factors. These can be classified as follows: machine tool and its controlling equipment, workpiece, fixtures / jigs, tools and environmental conditions.

In order to manage the problems of mechanical flexibility in general, an innovative scheme based on *inexact algorithms*, fuzzy logic, neural networks and neural fuzzy technique, has been developed. A *Spring Mounted Pen (SMP)* is used in the experiments to emulate the movement of an end-effector caused by flexible mechanical structures. Applying computer vision, it is possible to monitor the processing errors as well as generating on-line information to an Artificial Intelligence kernel. *2VMethod* together with a *Piecewise Error Compensation Algorithm* (*PEC Algorithm*) have been developed to automatically compensate for deviation due to mechanical flexibility. These methods have been reported in [1][2][3].

A new type of network architecture for constructing an intelligent neural fuzzy controller used in the system is developed to achieve fast training speed and higher accuracy. Through on-line self-learning process - the controller analyses the errors between the required shape and the resultant shape to make the necessary compensation. This paper is mainly concerned with the scheme used to design the novel neural fuzzy architecture. The 2VMethod and the PEC Algorithm are briefly reviewed. The results of using the intelligent kernel to create correcting patterns to overcome the problems of deformable structures are presented.

II. Compensating Algorithms

Two algorithms, the 2VMethod and the PEC Algorithm, are developed and used by the neural fuzzy kernel to determine the compensation pattern.

A. The 2VMethod

As depicted in Fig. 1, every *two consecutive coordinates* are analysed over the entire path. A straight line is connected between *Vector 1* and *Vector 2* (labelled in Fig. 1). The angles (θ1 and θ2) between the line (γ) and the X / Y coordinates are used to calculate the compensated vectors (δ(x),δ(y)). This correction is separated into two "energies" :

1) *Correcting pattern;* and 2) *Correcting amplitude.*

An on-line self-learning approach called the PEC Algorithm based on the inexact algorithms is applied to compute the correcting energies.

B. The PEC Algorithm

The learning procedure is divided into two steps. First, the Neural Fuzzy Engine (NFE One) reads the path-following-errors from the vision station and determines a possible correcting amplitude. Then a new detected path is fed into the NFE Two to compute the correcting pattern. Equation (1) describes the process of combining the correcting pattern and amplitude to create a compensation

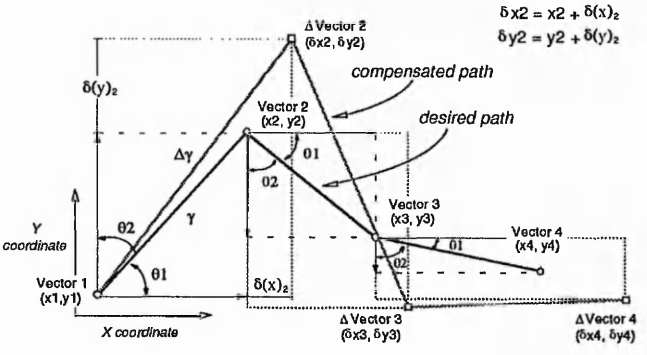


Fig. 1. Calculating new compensated vectors using the 2VMethod

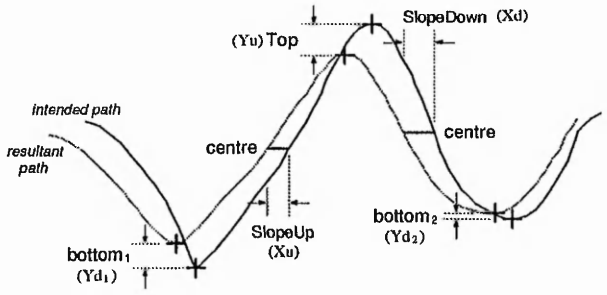


Fig. 2. Detecting the inaccuracy of the SMP following process

path.

$$PredictedSegment(i) = Pattern_{segment(i)} \cdot Amplitude_{path} \quad (1)$$

where i is the i^{th} segment of the path. The computation of the new compensated vector $(\Delta Vector2(\delta x2,\ \delta y2))$ is calculated by (2). Equation (3) presents the procedure for calculating the pattern of compensation.

$$\delta x(i) = x(i) + \delta(x)_i, \quad \delta y(i) = y(i) + \delta(y)_i \quad (2)$$

$$\begin{aligned} CompensationPath &= Vector(1) + \sum_{i=2}^{n} \Delta Vector(i) \\ &= \{x1, y1\} + \sum_{i=2}^{n} \{\delta x(i), \delta y(i)\} \end{aligned} \quad (3)$$

where i is the i^{th} segment of the path and n is the number of vectors in the path.

As the first processed frame is passed under the vision station, an image is taken and sent to the host system. A software recogniser is used to separate the intended path and the resultant path (see Fig. 2). The top, bottom and centre positions in both paths are taken to measure the inaccuracy of the SMP following process. The distances between these points within the different paths are calculated by NFE One which produces a suitable amount of amplitude for the correction. Combining the outputs from the NFEs (using (1)), the compensation path can be formed. The detailed description of applying the 2VMethod and the PEC Algorithm to create the compensation patterns can be found in [2][3]. The method used to construct the NFEs engaged in the project is discussed next.

III. Neural Fuzzy System

The concept of neural fuzzy theory has received much attention recently. Alternative methods of integrating neural networks and fuzzy logic have been proposed in the scientific literature. We have made use of neural networks for forming the required fuzzy membership functions and the fuzzy rules. The symmetric triangle and trapezoid membership functions are commonly used in fuzzy kernels. In our system, the shape of input membership function used is symmetric triangle, and the output membership function is symmetric trapezoid. The reason for choosing such combination of membership functions is that the system can produce the smoothest output pattern.

A proposed technique has been tested using the neuro-fuzzy architecture discussed in [4]. Applying this approach, the membership function of the fuzzy system is implemented using multi-layer pre-trained neural network which enables off-line learning of the function using a fully connected back-propagation feedforward network. Diagram (a), (b) and (c) in Fig. 3 show the results of the trained membership function used 0.01, 0.000001 and 10^{-12}, respectively, as the satisfactory error (δ) level to train the neuro-fuzzy kernel. It can be seen that the actual outputs of the kernel cannot completely match the desired pattern. The problem of using such membership function is that these imprecise output values will cause a certain amount of error during the inference procedure and compromise the performance of the intelligent kernel.

In order to improve the accuracy of the outputs. A new type of neural fuzzy architecture is introduced here. This system is divided into three functional blocks.

A. Generating input membership functions

A fully connected multi-layer network with sigmoid activation function, five neurodes in the first hidden layer and two in the second layer is used in the system. The supervised back-propagation training algorithm was engaged during the learning process. A primary element of the network for generating an input membership function is depicted in Fig. 4. From the neurodes in the second hidden layer of the network, (a) and (b) labelled in Fig. 4, both sides of the membership function can be yielded, respectively. These two signals are then fed into a filter (labelled (c)) which simply does the minimum (Λ) operation. Finally the entire membership function can be successfully generated and obtained from the output of the

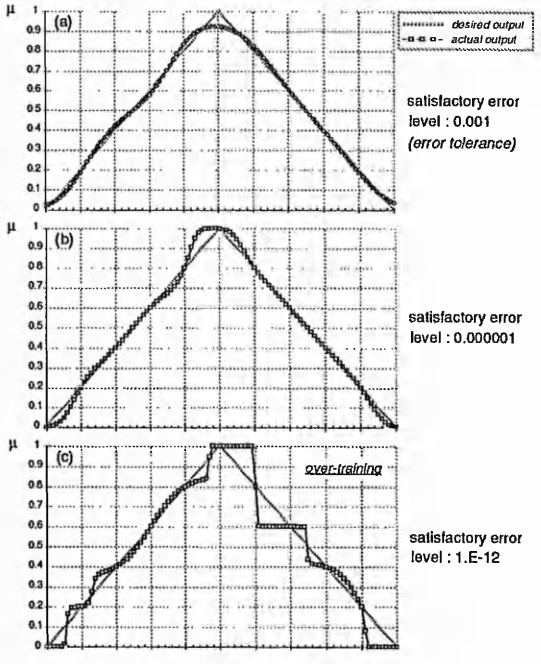


Fig. 3. Samples of the generated membership functions

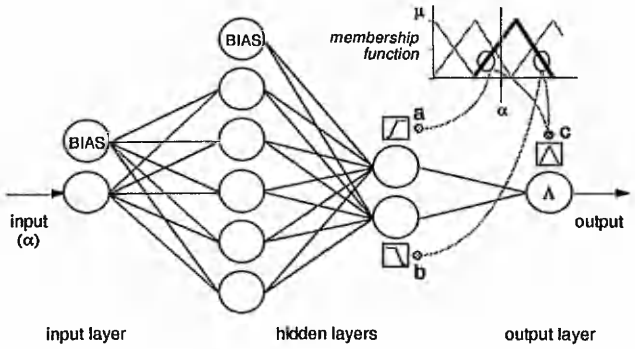


Fig. 4. Network structure for generating an input membership function

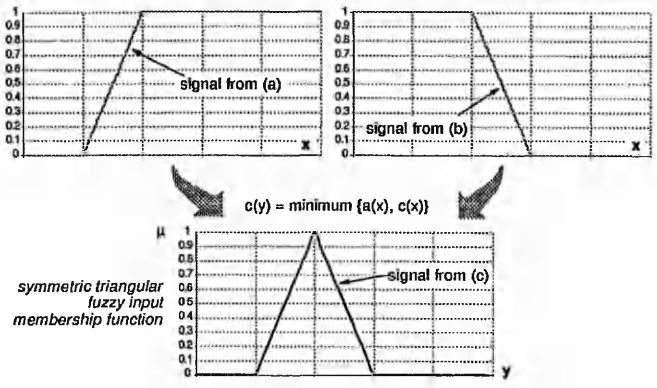


Fig. 5. Sample of input membership function generated by the NFE

network.

An example of using this network structure to produce a symmetric triangle membership function is illustrated in Fig. 5. In fact, by utilising the method the triangular, trapezoid or "bell" shaped input membership function can all be created.

B. Generating output membership functions

A network architecture as shown in Fig. 6 is designed to create an output membership function. A three-layered fully connected back-propagation network including three neurodes in the hidden layer and two in the output layer with sigmoid function is used. Only a small number of representative data set is required for training the

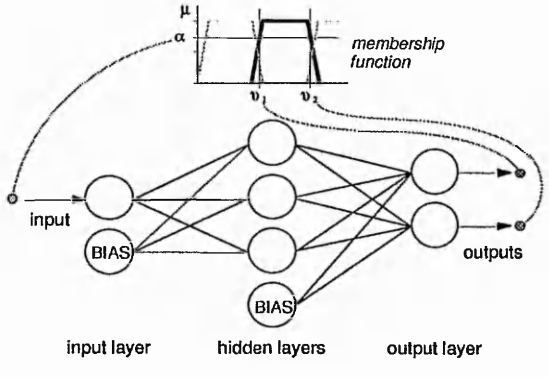


Fig. 6. Network structure for generating an output membership function

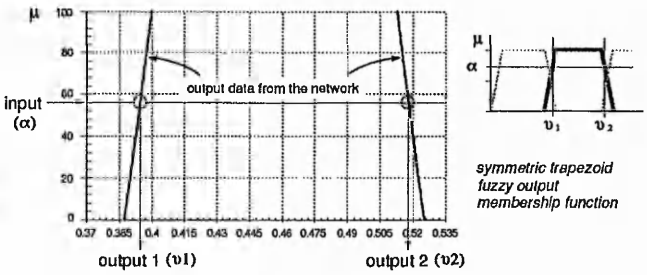


Fig. 7. Sample of output membership function generated by the NFE

network.

Fig. 7 shows the output data collected from the trained NFE. In the example a symmetric trapezoid output membership function is utilised to produce the training data set. Only less than ten seconds is required to successfully converge the network (δ: 0.00000001) in which an Intel 486 CPU running at 66 MHz is engaged. Other shapes of output membership functions can also created by applying this approach.

C. Connections of the rules

An example of connecting the rules of the NFE which has two-input and one-output nodes is presented in Fig. 8. Nine rules were derived from the Fuzzy Associative Memory (FAM) Bank. For instance, the following two conditions can have the same consequence:

IF input_1 = a AND input_2 = A THEN output = 1
IF input_1 = c AND input_2 = B THEN output = 1

A minimum (Λ) operation is used to merge neurodes (a) and (A) (indicated in Fig. 8). The same approach is applied to neurode (c) and (B). The output neurode (1) can then be acquired by joining nodes aA and cB using a maximum (V) operation. Equation (4) represents this procedure.

$$\text{Output 1} = V[\Lambda(a, A), \Lambda(c, B)] \qquad (4)$$

Using the same method, the rest of the rules can be arranged within the network.

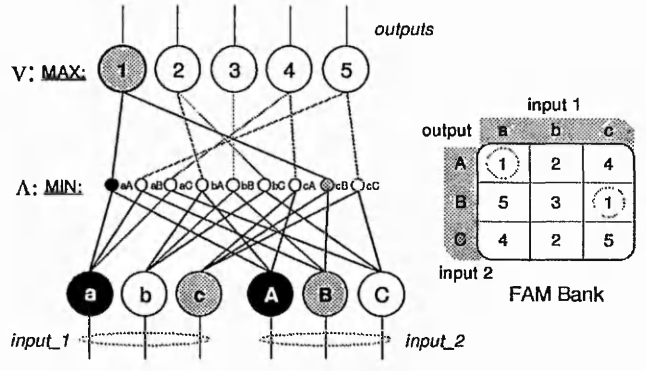


Fig. 8. Connections of the rules

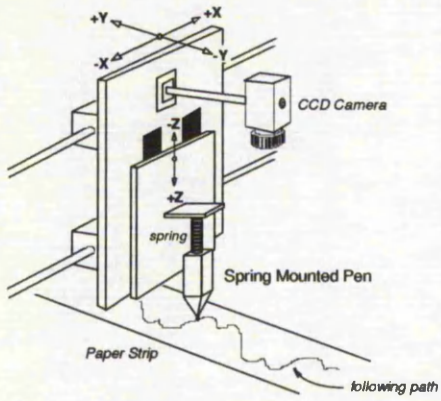


Fig. 9. Schematic diagram of the test rig

IV. Experimental Results

Various experiments have been carried out to evaluate the effectiveness of this algorithm. The SMP is connected with the Z axis of a CNC machine (Fig. 9). Fig. 10 shows two compensated paths generated by the NFEs during the learning process.

Once the compensated path is derived, this data is transferred to the controller of the test rig. The SMP is, then, driven to draw a line over the original path. Fig. 11 illustrates two samples of the SMP following process. Fig. 11-a presents the SMP following without adding any compensation. Fig. 11-b shows the sample after three learning frames where the compensation is fed-back to the controller. It can be seen that almost all the deviation can be successfully removed.

The experimental results indicate that using the algorithm developed the intelligent controller can deal with any regular and irregular shape of patterns and produce good results, even better than human operators. Fig. 12 presents a two-input / one-output neural fuzzy architecture developed. By applying the new network structure excellent results have been achieved and can perform better than that previously reported [4][5].

V. Conclusion

This paper introduced an intelligent controller using a neural fuzzy theory for handling flexibility of dynamic structures. The development of the algorithm is a novel approach to flexible sheet material processing and has further applications where modelling system behaviour characteristics, such as controlling a robot moving on a slippery surface or piloting a boat, is difficult. Furthermore, by relying on the intelligent controller

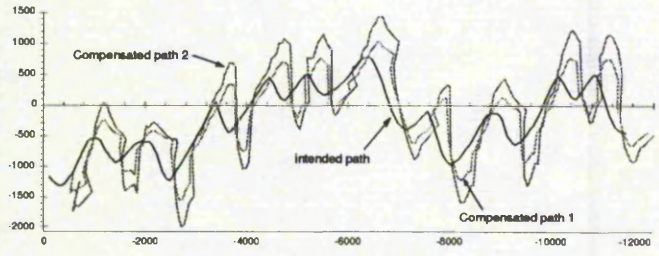


Fig. 10. Two compensated paths created by the NFEs

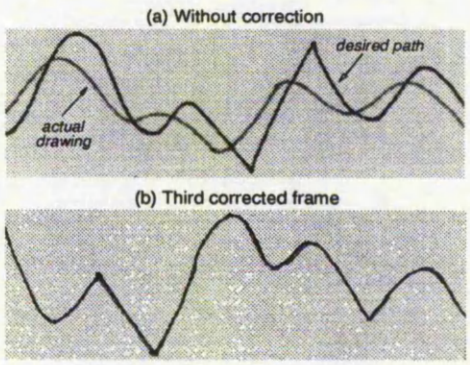


Fig. 11. Samples of SMP following applied the NFEs

together with the sensing station the system no longer needs to rely on accurate position feed-back. Backlash, joint flexibility and stick slip can potentially be compensated for by the controller. When characteristics of the mechanism change over time, such as component wear, temperature change, the controller can automatically make appropriate compensation.

References

[1] Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "Tightly coupled vision based control system using inexact algorithms", IEEE Second Asian Conference on Computer Vision, ACCV '95, Singapore, 1995.

[2] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Close coupling of pre- and post-processing vision stations using Inexact Algorithm", IS&T / SPIE's Symposium on Electronic Imaging : Science & Technology, USA, 1996.

[3] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Correction of errors due to flexibility of dynamic systems", 1996 IEEE International Conference on Robotics and Automation, Minnesota, USA, 1996.

[4] Leng, T. S.; Leng, N. S. and Tech, N. W. "Control of an inverted Pendulum using a neuro-fuzzy controller", IEEE: 0-7803-1223-6/93, 1993.

[5] Chow, M-Y and Goode, P. V. "Adaptation of a neural/fuzzy fault detection system", IEEE Proceedings of the 32nd Conference on Decision and Control, Texas, 1993.

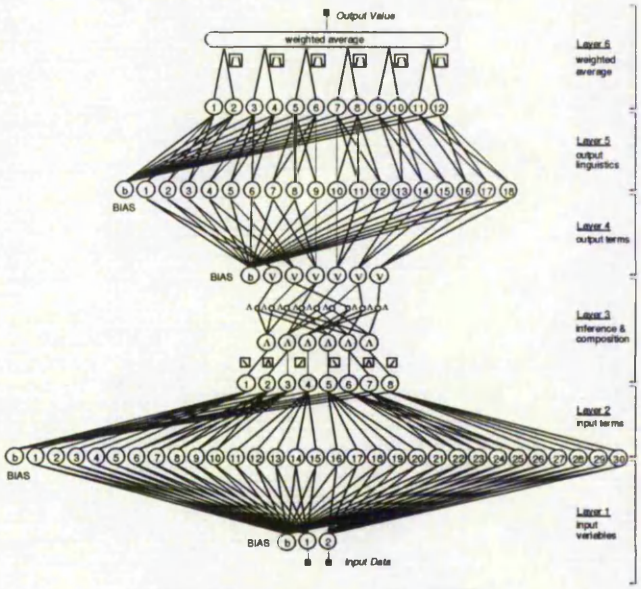


Fig. 12. Architecture of the NFE

Neural Fuzzy Based Self-Learning Algorithms for Handling Flexibility of Dynamic Structures

Chi-Hsien V. Shih, Nasser Sherkat, Peter Thomas
The Nottingham Trent University, Department of Computing
Burton Street, Nottingham NG1 4BU, United Kingdom

Abstract **– This paper describes a novel approach to tackle problems associated with handling flexibility of dynamic structures. A number of solutions to this problem have been developed by innovative combination of fuzzy logic and neural networks - *Neural Fuzzy Technique*. In order to emulate the deviation of an end-effector caused by flexibility, a *Spring Mounted Pen (SMP)* is designed and used in the experiments. *The Piecewise Error Compensation Algorithm (PEC Algorithm)* and the *Generic Error Compensation Algorithm (GEC Algorithm)* are devised to correct the deviations. Comparing the desired pattern and the actual output pattern, the vision based intelligent controller can automatically make appropriate compensation through an on-line self-learning process. Various experimental results indicate that applying the algorithms developed the intelligent kernel can compensate for flexibility and produce good results.**

I. INTRODUCTION

Manufacturing with high accuracy is influenced by a number of factors which can be classified as follows: machine tool and its controller, workpiece, fixtures / jigs, tools and environmental conditions. Vibrational errors and control induced errors that appear in a manufacturing system are normally ruled by these factors. Minimising the effect of such errors is usually costly. It would be desirable to rely on the intelligence of the controller to compensate for errors due to flexibility rather than resorting to costly processes of tightening the tolerances.

In order to tackle the problem of mechanical flexibility in general, a novel scheme based on fuzzy logic and neural networks has been developed and described here. A Spring Mounted Pen (SMP) is used in the experiments to emulate the movement of an end-effector caused by flexible mechanical structures (Fig. 1). Using machine vision station, it is possible to monitor a error as well as generating on-line information to the Artificial Intelligence kernel. This allows overcoming the problems of inaccuracy due to flexibility of dynamic structures. The developed algorithms are essentially trying to avoid using very complex sensors to monitor all the system and other environmental factors, such as those mentioned previously. Through a self-learning process - the intelligent kernel compares the difference between the desired shape and the resulting shape to make the appropriate compensation in real time.

For example a system which is subject to errors due to flexibility of the workpiece is a lace scalloping machine. A number of attempts have been made to automate the process of lace scalloping and quality inspection [1][2][3][4][5]. Work has been reported in using laser technology to cut deformable materials [6]. Although using laser reduces this deformation, distortion due to mechanical feed flexibility and misalignments persists. Changes in the lace pattern are also caused by the release of tension in the lace structure as it is cut. By using the developed algorithm, the problems in lace trimming can be overcome.

II. SYSTEM OVERVIEW

The host system receives an external video signal as well as displaying the captured image on the video monitor. The machine movement commands are generated and passed to the cutting mechanism and the transportation system (conveyor). A SMP, as depicted in

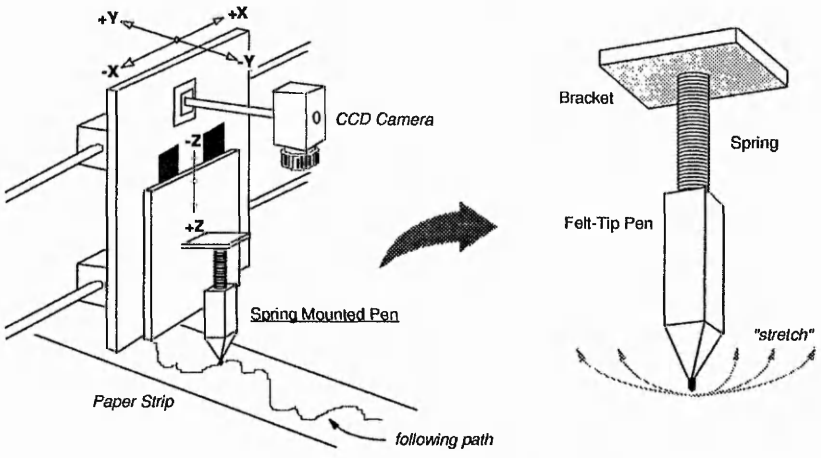


Fig. 1 Spring Mounted Pen (SMP) connected with the test rig

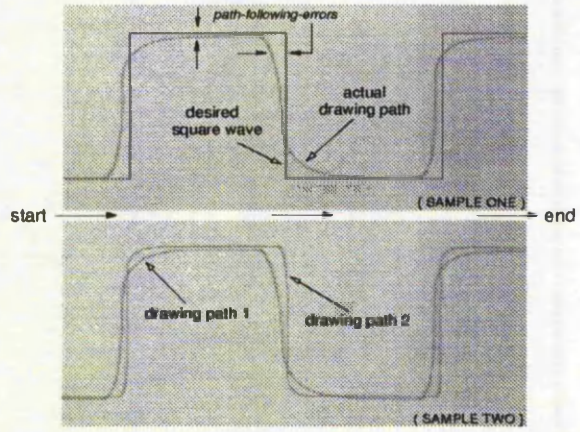


Fig. 2 Samples of square wave following process using the SMP

Fig. 1, is guided by the machine to draw a path on a paper strip to emulate the distortion of the deformable material due to the cutting forces caused by tactile cutting and feed misalignment. Fig. 2 represents the results of following a square wave using the SMP. Due to the inherent characteristics of the spring, the path-following-errors (PFEs) appear between the desired path and the actual target. Additionally, each time the pen is put in contact with paper, the axial load on the spring changes. This consequently causes the pattern generated by the SMP to alter (path1 and path2 indicated in Fig. 2).

It can be seen that the PFE occurs when the direction of the drawing is changed - the larger the angular variation of the path following, the larger is the error. The amount (magnitude) of the PFE generated depends on the *characteristic of the spring* engaged, the *pressure* on the SMP and the *friction force* between the tip of the pen and the paper. As any one of the system coefficients is altered, the result of the SMP drawing will be different.

III. INTEGRATING THE REMOTE SENSING BASED CONTROL

We have devised a vision based intelligent control system incorporating a neural fuzzy technique. As shown in Fig. 3, the PFE is detected and fed into A.I. Engine One which analyses the difference between the paths. It also decides the amplitude of correction for further processing. The vision station is triggered to capture a new frame of the desired path on paper. At this time, before the extracted curve is sent to the path generator, the segments of the extracted path are passed to A.I. Engine Two which determines the correcting pattern. Both the amplitude and the pattern are utilised to generate a predicted correcting path. Finally, the path generator uses the predicted path and the original path to produce the machine movement data (compensation path).

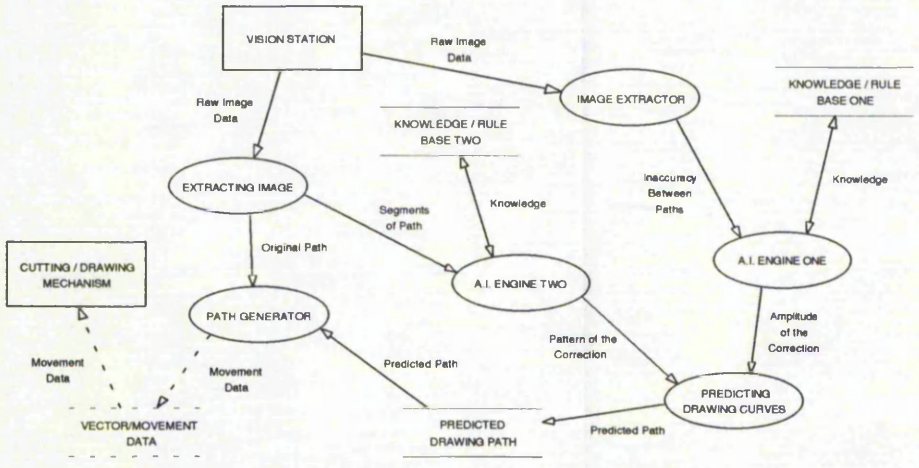


Fig. 3 The overview of the vision and motion control systems

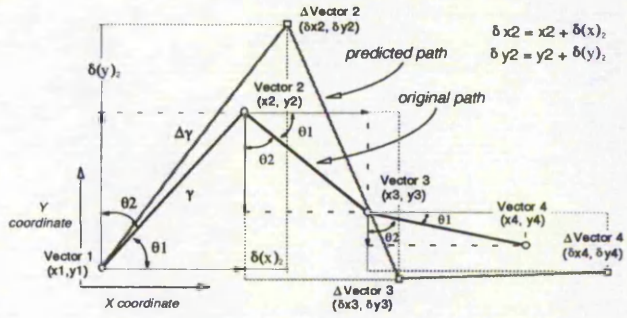


Fig. 4 Calculating new corrected coordinates

Two innovative schemes named the *Piecewise Error Compensation Algorithm (PEC Algorithm)* and *Generic Error Compensation Algorithm (GEC Algorithm)* form the basis of the compensation system. The detailed description of the PEC algorithm based on the neural fuzzy technique can be found in [7][8]. In the following section, the GEC algorithm using neural network approach is presented.

IV. THE GEC ALGORITHM

As depicted in Fig. 4, this approach considers a small portion of the desired path to predict the correction. Every two consecutive coordinates (a segment) are analysed over the entire path. As a straight line is connected from *Coordinate 1* to *Coordinate 2* (refer to Fig. 4), the angles (θ1 and θ2) between the line (γ) and the X / Y coordinates are used to compute the possible correcting energies ($\delta(x)$ and $\delta(y)$). The angles θ1 and θ2 are related to the each other - θ1 and θ2 are complementary. These two angles are passed to an A.I. engine to determine the correcting energies - δ(x) and δ(y). The prediction of the new estimated coordinate *(ΔCoordinate2(δx2, δy2))* is calculated by (1). Equation (2)

describes the procedure of computing the pattern of the predicted path.

$$\delta x(i) = x(i) + \delta(x)_i, \quad \delta y(i) = y(i) + \delta(y)_i \tag{1}$$

$$\begin{aligned} Predicted\ Path &= Vector(1) + \sum_{i=2}^{n} \Delta Vector(i) \\ &= \{x1, y1\} + \sum_{i=2}^{n} \{\delta x(i), \delta y(i)\} \end{aligned} \tag{2}$$

where i is the ith segment of the path and n is the number of vectors in the path. The correcting energies, $\delta(x)_i$ and $\delta(y)_i$, are also separated into two functions: *correcting pattern (CP)* and *correcting amplitude (CA)*. Equation (3) presents the relationship.

$$\delta(x,y)_i = CP_{segment(i)} \times CA_{path} \tag{3}$$

A. Detecting the Correcting Pattern

In order to detect the deviation caused by the spring, the SMP is driven to follow a template (a square wave) on paper. The image of this square wave is captured by the vision station. A software filter is developed to detected this captured image - six coordinates, such as those shown in Fig. 5 (a), can be obtained. This data is then transferred to the controller.

The SMP is driven to draw a second line on the paper. It is clear that the PFEs appear. The PFEs appear when the direction of the drawing changes, i.e. from direction −X to −Y, −Y to −X, −X to +Y and +Y to −X (see Fig. 5 (b)). There are four different types of deviation patterns that can be detected. They are labelled as nXnY, nYnX, nXpY and pYnX where 'n' denotes negative and 'p' means

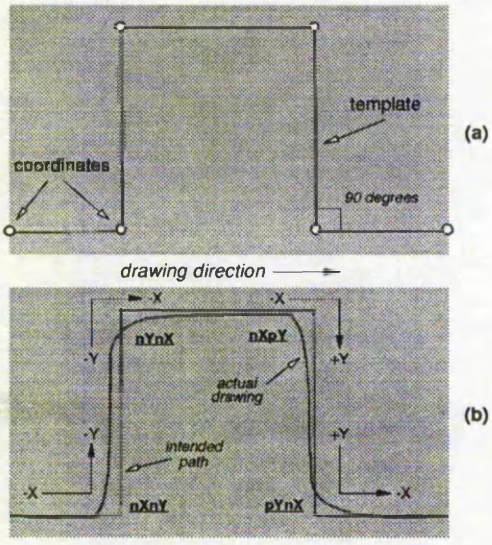


Fig. 5 (a) Vectorising the square wave; (b) Drawing a second path followed the template using the SMP

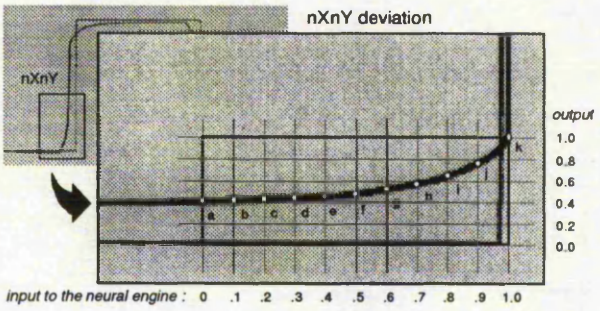


Fig. 6: Obtaining the training data set from a deviation pattern

positive.

The neural network approach is employed here to learn the correcting action from the deviation patterns (as shown in Fig. 6) for constructing the intelligent engine to produce the compensation patterns. Fig. 6 depicts the use of the nXnY deviation pattern to produce the learning data set for training a *neural engine (ANN Engine)*. Eleven data points (a, b, c,..., k labelled in Fig. 6) which are taken from experiments are chosen in this instance. As the training data is fed into the neural engine, after learning and updating procedure the trained neural engine can be used to generate a correcting pattern. The neural engine constructed in the system is a standard fully interconnected three layer back propagation network. Using the similar approach stated above, the learning data sets from nYnX, nXpY and pYnX deviation patterns can all be obtained and utilised for teaching the networks.

As mentioned previously, four different shapes of deviation patterns, i.e., nXnY, nYnX, nXpY and pYnX, can be detected from the result of following the template. Once the neural engine has successfully learned from these samples, the GEC algorithm is, then, used to calculate the compensated segments. Dissimilar to the PEC algorithm, instead of using only two correcting patterns [7][8], two sets of correcting patterns are used, i.e. $\{\delta(nXnY), \delta(nYnX)\}$ and $\{\delta(nXpY), \delta(pYnX)\}$. Depending on the direction (angle) of the line between two detected coordinates, one of the correcting pattern pair is assigned to the correcting energies $\delta(x)_i$ and $\delta(y)_i$. For example, if the angle of the line between two coordinates is less then 180 degrees (Fig. 7), then (4) is engaged by the GEC algorithm; or if the angle of the line is larger than 180 degrees then (5) is used.

$$\delta(x) = \delta(nXnY), \ \delta(y) = \delta(nYnX) \tag{4}$$

$$\delta(x) = \delta(nXpY), \ \delta(y) = \delta(pYnX) \tag{5}$$

where $\delta(nXnY)$ is the correcting pattern generated by the neural engine which uses $nXnY$ deviation pattern as the learning data. Since the cutting mechanism employed in the project is controlled to move from +X to −X direction, we only need to consider the angles of the coordinates which are larger than 90 degrees and less than 270

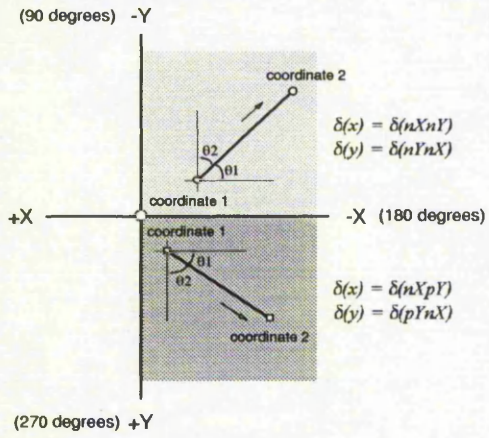


Fig. 7 Depending on the direction (angle) of the path, two sets of correcting patterns can be chosen to assign for the GEC algorithm

degrees. As the angles θ1 and θ2 (labelled in Fig. 7) are detected, this information is normalised into the range of [0, 1]. Additionally, this normalised data is passed onto the trained neural engine in order to produce the correcting energies $\delta(x)_i$ and $\delta(y)_i$ which can be used to create the correcting pattern. Fig. 8 represents this procedure.

B. Detection of Correcting Amplitude

Once the correcting patterns are obtained, the next step is to determine the amount of correcting amplitude required. As already stated when the Z axis of the test rig is reset, three system parameters of the SMP are altered (refer to Section II). This results in changing the magnitude of the PFEs (Fig. 9). In order to measure the maximum amount of PFEs that can be produced by the SMP, the actual length of the deviation is calculated.

Fig. 10 depicts an example of calculating two maximum deviations caused by the SMP. The vision station is used to detect the length of L1 (or L2). The actual length of L1 is then transformed into the machine control unit which is 40 steps / mm. As an illustration, if L1 is measured as 8.9 mm, the maximum machine control units that can be added in the original path in $\delta(nXpY)$ side is 356 steps (8.9 mm × 40 steps / mm).

C. The Correction Process

As the test rig is set up, the SMP is driven to draw a square wave on paper. Four deviation patterns are taken to create the learning data sets for training the neural network kernel. Besides, four different maximum deviations (*nXnY*, *nXpY*, *nYnX* and *pYnX*) are detected and stored in a configuration file. A new frame of the desired drawing path is grabbed by the CCD camera and analysed. The detected pattern (original path) is then vectorised and fed into the trained neural engine. By applying (1), (2) and (3), the correcting patterns and amplitudes as well as the original detected path are used to create the compensated path. This vectorised data is, finally, transferred to the controller.

V. EXPERIMENTAL RESULTS

A working prototype is constructed. A number of experiments have been carried out to evaluate the effectiveness of this algorithm. Fig. 11 illustrates the processes of correcting the deviation by applying the GEC algorithm using the neural network approach. As depicted, almost all the PFEs can be successfully removed after one frame of training procedure. Two samples of SMP following process and their compensated patterns can be seen in Fig. 12 and Fig. 13. The test rig developed by the authors is illustrated in Fig. 14.

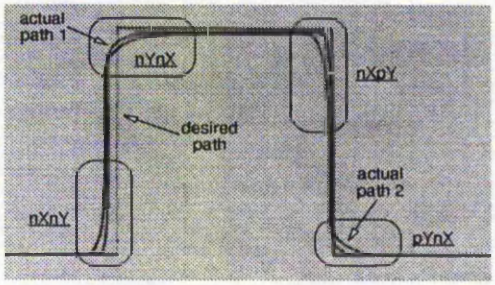


Fig. 9 Sample of template following using different system setting

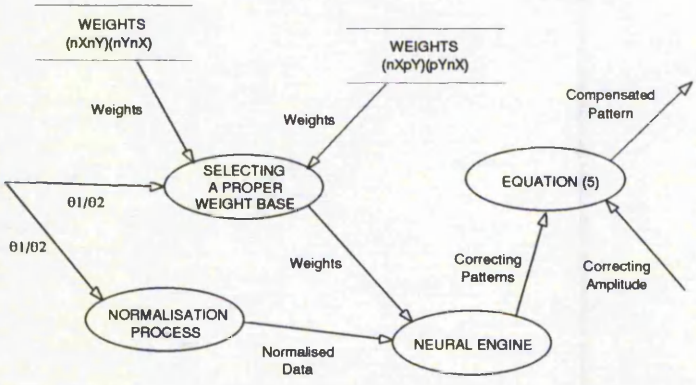


Fig. 8 Use of a neural engine to compute the compensated pattern

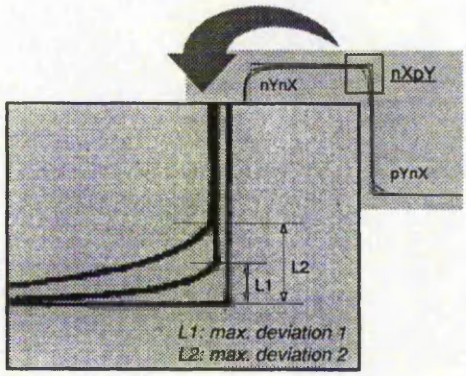


Fig. 10 Example of calculating maximum deviations

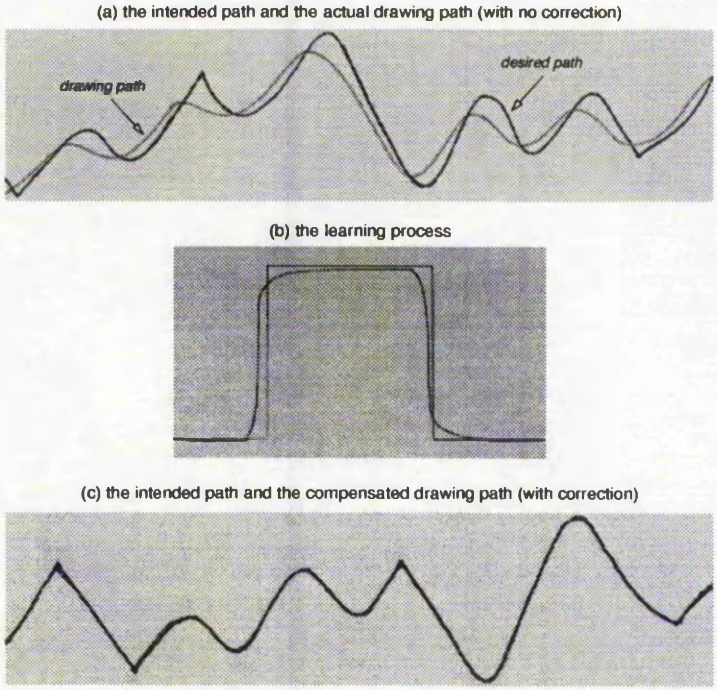


Fig. 11 The processes of correcting the PFEs using the GEC algorithm

Various experimental results indicate that by relying on the algorithms developed the system can deal with any regular and irregular paths and produce excellent results, much better than a human operator.

VI. CONCLUSION

We have presented attempts to develop a vision based intelligent control system for compensating errors due to flexibility of dynamic structures. The development of the system is an innovative approach to flexible material processing and has further applications where modeling system behaviour characteristics is difficult. Such systems can range from controlling a robot moving on a slippery surface, driving a car on snow or piloting a boat, etc. Furthermore, by relying on the intelligent software engine

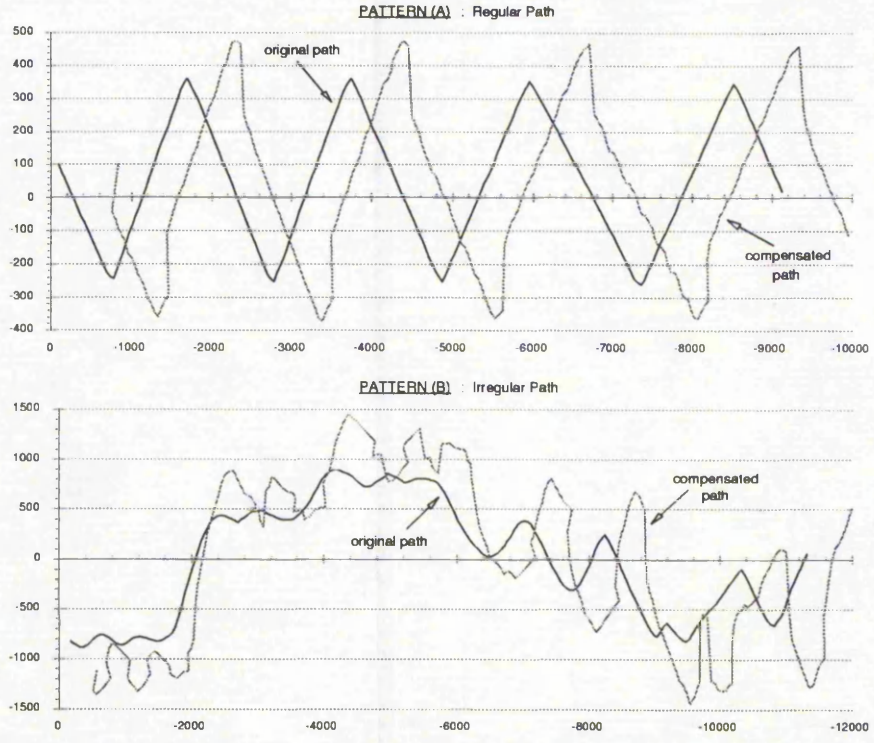


Fig. 12 Two compensated paths generated by the neural engine using the GEC algorithm

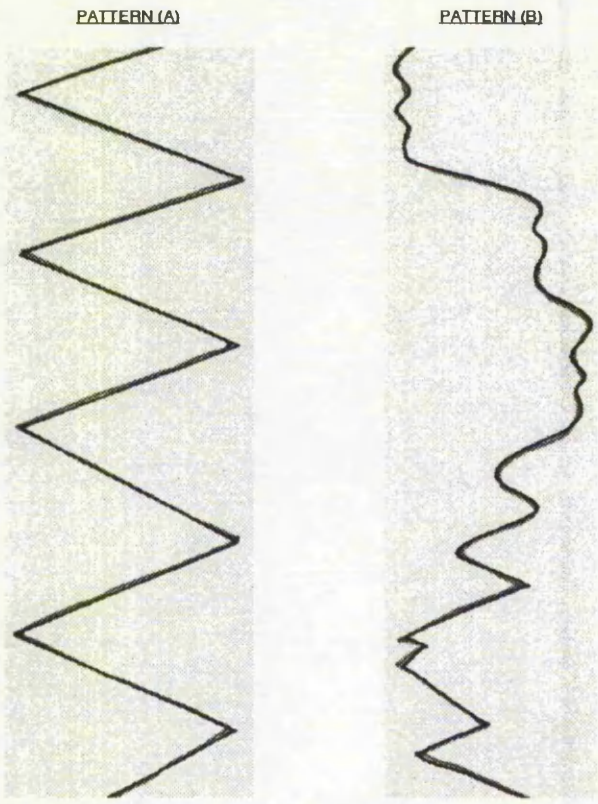


Fig. 13 Samples of SMP following using the GEC algorithm

together with the vision system the controller no longer needs to rely on accurate position fed-back from the sensors (encoders). Transmission backlash, joint flexibility, poor feedback and stick slip can potentially be compensated for by the controller. While the characteristics of the mechanism change over time, such as component wear, temperature change, and/or cheaper materials used in construction, the controller can automatically make appropriate compensation. An industrially sponsored programme of work has just commenced to develop a commercial machine controller based on the developed principle.

It is sensible to anticipate that computer hardware will decrease continuously in cost while increasing in performance. In contrast, mechanical hardware costs are much likely to stay in line with inflation in the future years. Consequently, it is reasonable to, where possible, make a shift from mechanical hardware to computer with the associated intelligent software kernel in automated industrial applications.

VII. ACKNOWLEDGMENTS

This work has been carried out in collaboration with Axiomatic Technology Ltd. and Pacer Systems Ltd.

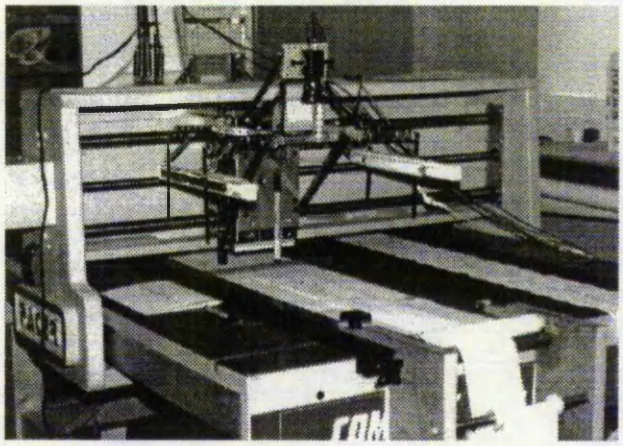


Fig. 14 Prototype of a vision based intelligent control station

REFERENCES

[1] Sherkat, Nasser; Shih, Chi-Hsien V. and Thomas, Peter. "A fuzzy reasoning rule-based system for handling lace pattern distortion", *IS&T / SPIE's Symposium on Electronic Imaging : Science & Technology*, vol. 2423, California, USA, 1995, pp. 323-333.

[2] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Real-time tracking of lace stretch using machine vision", *IEE Fifth International Conference on Image Processing and Its Applications*, no. 410, Edinburgh, U.K., 1995, pp. 687-691.

[3] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Automation of lace cutting using real-time vision", *8th International Congress on Condition Monitoring and Diagnostic Engineering Management*, Canada, 1995, pp. 245-252.

[4] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "An automatic lace trimming process using real-time vision", *Journal of Real-Time Imaging*, April 1996.

[5] Norton-Wayne. L. "Inspection of lace using machine vision", *Computer Graphics Forum*, Vol.10, 1991, pp. 113-119.

[6] King, T. "Real-time tracking of patterns on deformable materials using DSP", *IEE International workshop on systems engineering for real-time applications*, Royal Agricultural College, Cirencester, U.K, 1993, pp. 178-183.

[7] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Close coupling of pre- and post-processing vision stations using inexact algorithms", *IS&T / SPIE's Symposium on Electronic Imaging: Science & Technology*, California, USA, 1996.

[8] Shih, Chi-Hsien V.; Sherkat, Nasser and Thomas, Peter. "Correction of errors due to flexibility of dynamic systems", *1996 IEEE International Conference on Robotics and Automation*, USA, 1996.

Memorandum 1

Memorandum 2

Memorandum 3