

FOR REFERENCE ONLY

26 JUL 2002

40 0720627 7



ProQuest Number: 10183399

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



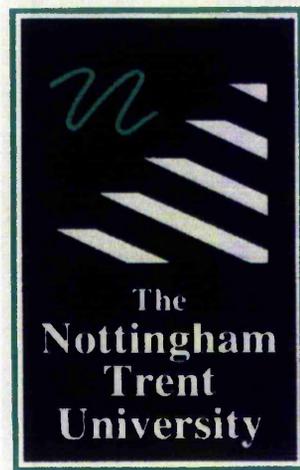
ProQuest 10183399

Published by ProQuest LLC (2017). Copyright of the Dissertation is held by the Author.

All rights reserved.

This work is protected against unauthorized copying under Title 17, United States Code  
Microform Edition © ProQuest LLC.

ProQuest LLC.  
789 East Eisenhower Parkway  
P.O. Box 1346  
Ann Arbor, MI 48106 – 1346



# **Primitive-based segmentation for triangulated surfaces**

**Rainer Sacchi**

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

September 2001

The Nottingham Trent University (TNTU)  
in collaboration with  
the Institute of Applied Computer Science (IAI)  
at the Research Centre Karlsruhe (FZK), Germany

---



**Forschungszentrum Karlsruhe**  
Technik und Umwelt

---

**Institut für Angewandte Informatik (IAI)**

## Abstract

Numerous fields of applications require a digital model to be produced from a physical object, for example in computer-aided design of appliance casings, manufacture of engineering components and virtual reality. A digital description allows changes in design and manipulation of the data, where it would be, for example, too expensive, too dangerous, or too time-consuming to do the same in the physical world. In order to obtain data from the physical object, coordinate measuring devices record data either by contact-less or by contact measurement, and in many cases the result yields an unstructured point cloud. Before the surface of a digitised object can be manipulated interactively, it must be reconstructed from the (possibly unstructured) set of points. With structured points the generation of a triangulated surface is relatively straightforward. To generate an initial triangulated surface out of the unstructured point cloud sophisticated methods have been established. The present project aims to achieve the next important step in surface reconstruction, namely to segment the triangulated surface into parts of simple geometric primitives, in particular of the following: plane, sphere, cylinder, cone, and torus. Such segmentation enables engineers to manipulate data for design purposes more quickly, because connected point sets, rather than individual points, will be affected. Subsequently the data can be used for "rapid prototyping", i.e. the manufacture of a physical model from a digital description.

In order to obtain a segmentation of a triangulated surface the approach for the extraction of geometric primitives used in this project has been based on a "region growing" method. It attempts to grow small initial seed regions satisfying a "homogeneous shape" criterion within a given tolerance. Each time the growing process yields a sufficiently large connected set of triangles a new segment of a geometric primitive with its corresponding characteristic parameters and boundary curves is identified. An additional source of shape information about triangulated surfaces is an estimate of curvature for each triangle. Curvature information allows the selection of appropriate seed regions, and it allows good initial estimates of characteristic parameters to be found. This is important because the growing process under preservation of shape involves numerical optimisation, whereby the initial characteristic parameters are adjusted as the region grows.

Methods of curvature estimation for triangulated surfaces have been investigated. Curvature estimation algorithms for triangulated surfaces have been developed and evaluated for both synthetic data (for which curvature values are known) and "real" data. They compare favourably with other curvature estimation algorithms suitable for discrete data. A formula for the sign of curvature has been found in the literature to give sometimes a wrong result and an appropriate correction has been suggested.

Algorithms for region growing have been established which are based on the curvature estimates obtained. Techniques have been developed for determination of initial characteristic parameters for planes, spheres, cylinders, cones, and tori using the estimated curvature, when only very small seed regions are available. Further work has established how characteristic parameters of segments of geometric primitives can be adjusted by region growing formulated as a minimax optimisation problem.

A fast method for the extraction of planar patches on a triangulated surface has been developed which is faster than the numerical approaches needed for the more complicated geometric primitives. This extraction employs a new, simple geometric method that exploits the asymptotic behaviour of an "expanded triangle" used to represent the plane. This method cannot only be applied to triangulated surfaces but also to any data representation that provides adjacency information. Results from the extraction involving all types of the above geometric primitives have been evaluated on "real" data. For many simple objects successful segmentation has been achieved and it is expected that further refinement of the developed algorithms will enhance their performance.

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

---

---

To my parents,  
my friends,  
and Julia

---

---

## Acknowledgements

The Author expresses his deep gratitude to many individuals who granted him support in the composition of this thesis. First of all, I am grateful to my first supervisor at the Department of Computing and Mathematics (DoCM) at TNTU, Dr. Janet Poliakoff, for her constructive criticism to my work, for her inspiring discussions, her continuous support that have helped me again and again to avoid the “pitfall of the obviousness”, and her patient attempts to teach a curious “foreign worker” the subtle nuances of the English language. I am also very grateful to my second supervisor and the leader of the Real Time Machine Control group at the DoCM, Prof. Dr. Peter Thomas, for his continuous support with respect to a modern well-suited project equipment, financial funding, and his constructive criticism regarding the design of my thesis. Moreover, I say a big thank you to my external advisor at the IAI, Dipl.-Ing. Karl-Heinz Häfele, for motivating this fascinating work, his steady and constructive support, for providing an excellent organisational infrastructure in an up-to-date research institution, and for establishing an edge-cutting software system (POMOS) that served as the appropriate implementation framework for this project.

Many thanks are due to Dr. Peter Kohlhepp at the IAI for his helpful suggestions concerning the structure of Chapter 2 and for many valuable discussions. I also appreciate the assistance of the students of the DoCM during their placement at the IAI, Mr. Philip Lakin and Mr. Sharaz Mushtaq, for pushing this project forward with their dedicated programming work.

Because the Author needed to commute between two institutions, he thanks the following individuals for their kind support concerning the organisational infrastructure and for treating him as a colleague though he was torn between two worlds: Mr. Emmanouil Hatiris, Dr. Peyman Kabiri, Dr. Paul Orton, Dr. Adrian Stout, Mr. Kok-Hoe Wong (all at DoCM), Dipl.-Ing. Markus Dyckerhoff, Prof. Dr. Daniel Fischer, Dipl.-Inf. Achim Göbell, Dipl.-Inf. Helmut Knüppel (all at IAI), and many more.

Special thanks are directed to Prof. Dr. Andrzej Bargiela (DoCM), Prof. Dr. Georg Bretthauer (IAI), and Dr. Horst Haffner (IAI) for additional financial funding.

Last but not least the Author thanks all those who provided suitable test data to this project. These are in particular Dipl.-Ing. Karl-Heinz Häfele (IAI), Prof. Dr. Josef Hoschek (Technische Hochschule Darmstadt, Germany), Dr. Gerhard Roth (National Research Council of Canada, Ottawa), and Dr. Rayman Bardell (formerly at the Department of Electrical Engineering in The Nottingham Trent University).

# Table of contents

Glossary of terms .....	x
List of symbols .....	xv
<b>1 Introduction .....</b>	<b>1 - 1</b>
1.1 Surface modelling .....	1 - 2
1.2 Surface reconstruction by reverse engineering .....	1 - 4
1.2.1 Scanning of a physical object .....	1 - 5
1.2.2 Registering multiple views into a single data set .....	1 -12
1.2.3 Model generation .....	1 -13
1.3 Significance and examples of triangulated surfaces in CAD systems .....	1 -15
1.4 Motivation for surface segmentation .....	1 -16
1.5 POMOS - the implementation framework .....	1 -18
1.6 Objectives of this project .....	1 -19
<b>2 Survey of surface segmentation .....</b>	<b>2 - 1</b>
2.1 Surface representation and classification .....	2 - 2
2.2 Previous work in surface segmentation for range images .....	2 - 5
2.2.1 The segmentation problem .....	2 - 7
2.2.2 Top-down approaches .....	2 -10
2.2.3 Bottom-up approaches .....	2 -11
2.2.4 Clustering techniques .....	2 -12
2.2.5 Special techniques and additional remarks .....	2 -14
2.3 Curvature-based segmentation .....	2 -15
2.3.1 Explanation of curvature for smooth surfaces .....	2 -15
2.3.2 Application of curvature to the segmentation task .....	2 -19
2.4 Region growing as an efficient bottom-up approach for range images and triangulated surfaces .....	2 -21
2.4.1 Estimates of characteristic parameters .....	2 -22
2.4.2 Surface fitting.....	2 -23
2.4.3 Segment extraction .....	2 -26
2.5 Open problems .....	2 -29
2.6 The approaches used in this project: curvature estimation and region growing .....	2 -31
2.6.1 Representation of geometric primitives for surface segmentation .....	2 -31
2.6.2 Problem formulation .....	2 -32
<b>3 Curvature estimation for triangulated surfaces .....</b>	<b>3 - 1</b>
3.1 Previous methods of curvature estimation .....	3 - 1
3.2 Derivation of the novel NEN method .....	3 - 4
3.2.1 Defining interpolated and compensated normals .....	3 - 6
3.2.2 Estimation of principal curvature values and directions ...	3 -10
3.3 Appropriate synthetic and "real" test data .....	3 -16
3.4 Comparison of curvature estimation methods .....	3 -19
3.5 Application of the NEN method to non-synthetic data .....	3 -24
3.6 Summary .....	3 -29

<b>4</b>	<b>Region growing for geometric primitives on a triangulated surface .....</b>	<b>4 - 1</b>
4.1	Problem definition in the context of region growing .....	4 - 3
4.2	The modified region growing algorithm for triangulated surfaces .....	4 - 4
4.3	The estimation of characteristic parameters from seed triangles .....	4 - 8
4.3.1	Plane .....	4 - 9
4.3.2	Sphere .....	4 -10
4.3.3	Cylinder .....	4 -10
4.3.4	Cone .....	4 -11
4.3.5	Torus .....	4 -15
4.4	Methods of parameter optimisation .....	4 -20
4.4.1	Parameter optimisation employing a Genetic Algorithm ..	4 -21
4.4.1.1	Principles of Genetic Algorithms .....	4 -22
4.4.1.2	Surface fitting with a Genetic Algorithm .....	4 -24
4.4.2	Parameter optimisation employing Direct Search .....	4 -27
4.4.3	Parameter optimisation by a combined method .....	4 -28
4.5	Algorithmic parameters and their settings .....	4 -28
4.5.1	General algorithmic parameters .....	4 -29
4.5.2	Algorithmic parameters specific for a Genetic Algorithm ..	4 -30
4.5.3	Algorithmic parameters specific for the Direct Search method .....	4 -31
4.6	Examples of segmentation results .....	4 -31
4.6.1	Segmentation of a hemisphere attached to a plane .....	4 -33
4.6.2	Segmentation of a mechanical part .....	4 -35
4.6.3	Segmentation of a watering can .....	4 -39
4.6.4	Segmentation of a toy boat .....	4 -44
4.7	Discussion of methods and results .....	4 -48
4.7.1	Extraction of geometric primitives by modified region growing .....	4 -48
4.7.2	Estimation of characteristic parameters .....	4 -49
4.7.3	Robustness of the algorithmic parameters .....	4 -50
4.7.4	Evaluation of the segmentation results .....	4 -54
4.8	Summary .....	4 -56
<b>5</b>	<b>Improved extraction of planar segments .....</b>	<b>5 - 1</b>
5.1	The basic method and its problems .....	5 - 2
5.2	Improvements of the basic method .....	5 - 9
5.2.1	Analysis of data accuracy by local fitting of a plane .....	5 -10
5.2.2	Extracting planes using an adaptive tolerance .....	5 -11
5.2.3	Pseudo-random seed triplet selection .....	5 -12
5.3	Extraction results .....	5 -13
5.4	Summary .....	5 -17
<b>6</b>	<b>Discussion .....</b>	<b>6 - 1</b>
6.1	Curvature estimation .....	6 - 1
6.2	Region growing and determination of characteristic parameters for geometric primitives .....	6 - 1
6.2.1	Segmentation problem definition .....	6 - 2
6.2.2	Initial estimation of characteristic parameters for geometric primitives .....	6 - 3
6.2.3	Successive parameter optimisation for surface fitting .....	6 - 4
6.2.4	Segmentation results .....	6 - 6

6.3	Fast extraction of planar segments .....	6 - 7
6.4	Implications and future work .....	6 - 9
6.4.1	From a segmented to a parameterised surface .....	6 -10
6.4.2	Potential improvements of the segmentation .....	6 -10
6.4.3	The need for re-triangulation of a triangulated surface ...	6 -12
7	<b>Conclusions</b> .....	7 - 1
8	<b>References</b> .....	8 - 1
<b>Appendix</b>		
A	<b>Modelling point-surface distances for each of the geometric primitives used in this project</b> .....	A - 1
A.1	The distance of a point to a plane .....	A - 2
A.2	The distance of a point to a sphere .....	A - 3
A.3	The distance of a point to a cylinder .....	A - 4
A.4	The distance of a point to a cone .....	A - 5
A.5	The distance of a point to a torus .....	A - 6
B	<b>Derivation of curvature formulae</b> .....	A - 8
B.1	The DN curvature formula .....	A - 8
B.2	The interior point formula for a point on a circle and a non-circle point .....	A - 9
B.3	The interior point formula for two non-circle points .....	A - 9
C	<b>Alternative methods for estimating the characteristic parameters of cylinders, cones, and tori</b> .....	A-12
C.1	Cylinder .....	A-12
C.2	Cone .....	A-13
C.3	Torus .....	A-14
D	<b>Publications</b> .....	A-15
E	<b>Hardware used for this project</b> .....	A-16

## Glossary of terms

*2½D property*: property of surface data such that the data allows for a *parametric equation*; this property is characteristic for data in a *range image*.

*Adaptive tolerance*: a tolerance value automatically obtained from analysing the measurement errors in a data set in contrast to a predefined tolerance that is determined, for example, by user-interaction.

*Algorithmic parameters*: the parameters of the *segmenter* presented in this thesis that affect the segmentation result (see Section 4.5).

*Characteristic parameter estimation*: the process to determine *characteristic parameters* from a *seed region* in a given data set.

*Characteristic parameters*: a set of parameters uniquely characterising a specific instance of a geometric primitive or a parameterised surface.

*Compensated centre* (of a triangle): a calculated point within a triangle obtained from weighted averaging of the vertices (see *compensated normal*).

*Compensated normal* (of a triangle): a calculated normal of a triangle that aims to be the “true” surface normal at the *compensated centre* of a (possibly smooth) *virtual original*.

*Concurrent segmentation*: the fitting of different shapes to the same data set which may cause ambiguities in the interpretation of the data.

*Design*: the activity of creating (in most cases interactively) a geometric shape.

*Directional (optimisation) methods*: class of optimisation methods building up a sequence of points in order to find an optimum of a function  $f: \mathbb{R}^n \rightarrow \mathbb{R}$ .

*DN (curvature) formula*: formula for the estimation of curvature from two points on a curve or surface and its corresponding normal vectors; this formula is explained in greater detail in Equation (3.1).

*Edge detection*: a technique that aims to track areas of (locally) maximal curvature; this technique can be used to segment a surface by “cutting” it up along the detected edges.

*Encoding of chromosomes* (for a genetic algorithm): the encryption of genetic information of an individual into a numeric sequence (see Section 4.4.1.1).

*Expanded triangle (ET)*: a virtual triangle providing an improved approximation of a planar region compared to a *reference triangle (RT)* while extracting planes from a given data set; in general this triangle is larger than the *RT* and can therefore be used to update the *RT* (see Chapter 5).

*Explicit representation* (of a surface): a continuous mapping of the form

$z = g(x, y)$  where  $g: \Delta \rightarrow \mathfrak{R}$  and  $\Delta \subseteq \mathfrak{R}^2$  is a connected domain.

*Extracted average distance*: a statistical value for the *goodness of fit* that is obtained from averaging the distances of data points to a small plane fitted through these points.

*Genetic Algorithm (GA)*: a *non-directional optimisation method* based on the evolutionary principles of *mutation*, *recombination*, and *selection* in form of the “survival of the fittest” information that is encoded in chromosomes of multiple individuals (see Section 4.4.1.1 for details).

*Geometric modelling*: a collection of techniques or tools that may be used in both design and modelling.

*Goodness of fit*: a measure for the fit of data points to a given surface (for example, the least-squares sum of their Euclidean distances to the surface).

*Implicit equation* (of a surface): a continuous mapping of the form  $f: \Delta \rightarrow \mathfrak{R}$ ,  $f(x, y, z) = 0$  that defines a surface in a domain  $\Delta \subseteq \mathfrak{R}^3$ .

*Initial population* (of a genetic algorithm): the initial pool of individuals that carry genetic information of which the best are selected for further reproduction (see Section 4.4.1.1).

*Interior point curvature formula*: generic formula for the estimation of curvature on a polygon from two points (of which at least one is not a polygon vertex) and its corresponding normal vectors; this formula is explained in greater detail in Equation (3.6).

*Interpolated normal*: a calculated normal at a vertex of a triangulated surface obtained from weighted averaging of normals of triangles that meet in this vertex; an interpolated normal aims to be the “true” surface normal of a (possibly smooth) *virtual original*.

*Locally parameterisable*: a property of a surface  $S$  stating that the neighbourhood of an arbitrary point on  $S$  can be mapped bijectively onto a bounded, connected subset of a 2D plane “without holes”.

*Missed segment*: a segment that is present in a data set although it has not been identified.

*Modelling*: the activity of constructing a mathematical or computer model from the description of a shape, usually given in the form of an engineering drawing or stored in a wire-frame format.

*Mutation*: the process of changing genetic information randomly (e.g. affected by radiation) as part of the natural evolution (see Section 4.4.1.1).

*“Nine-fold evaluation of normals” (NEN) method*: generic, discrete method to estimate principal curvature values on a triangulated surface by applying the *interior*

*point curvature formula* to nine pairs of points and corresponding normal vectors (normals).

*Noise segment*: a physically non-existing segment resulting from the erroneous decomposition of a data set.

*Noisy data*: data that is notably affected by measurement errors.

*Non-directional (optimisation) methods*: class of optimisation methods that find an optimum of a function  $f: \mathcal{R}^n \rightarrow \mathcal{R}$  by evaluating  $f$  in multiple points and by comparing the resulting values to each other.

*Over-segmentation*: the decomposition of a data set such that segments of homogeneous shape are split into multiple instances of the same or of a different shape type.

*Parameter optimisation*: the process of (*surface*) *fitting*, applied to the *characteristic parameters* describing the surface.

*Parametric equation* (of a surface): a continuous mapping of the form  $f(u,v) = (x(u, v), y(u, v), z(u, v))$  where  $f: \Delta \rightarrow \mathcal{R}^3$  and  $\Delta \subseteq \mathcal{R}^2$  is a connected domain.

*Principal curvature direction*: direction in a point  $\mathbf{P}$  on a surface  $S$  in which  $\mathbf{P}$  corresponds to a *principal curvature* (value).

*Principal curvature (values)*: values denoting the maximum and the minimum curvature in a point  $\mathbf{P}$  of a surface  $S$  which can be obtained by approximating the surface in  $\mathbf{P}$  in a specific direction on  $S$  by an osculating sphere; each curvature value can then be obtained as one over the sphere's radius.

*Principal radius of curvature*: the reciprocal value of the corresponding *principal curvature*.

*Range image*: a viewpoint-dependent image of a physical object consisting of (often integer) distances from points on the object's surface to the viewpoint; range images are obtained from *scanning systems* yielding a *parametric equation* of the surface of a scanned object.

*Recombination*: the process of combining encoded chromosomes of two parental genes in order to obtain a child gene (see Section 4.4.1.1).

*Reference triangle (RT)*: a virtual triangle used to approximate a planar region while extracting planes from a given data set; distances of candidate points that could be added to the current planar region are measured relative to this triangle (see Chapter 5).

*Region*: a subset of a *regular triangulated surface* (see Section 4.1) or a subset of data points approximating a surface (see Chapter 5).

*Region growing*: a bottom-up segmentation approach attempting to grow a homogeneous region from a seed region by successive adding of data points.

*Registration*: the alignment of two or more range images into a common coordinate system.

*Regular triangulated surface*: a triangulated surface that meets the following three requirements: 1) each of its triangles has positive area; 2) the neighbourhood of each non-border point  $\mathbf{P}$  is *locally parameterisable*; 3) for every pair of surface triangles exists a sequence of edge-connected triangles that begins in one triangle of the pair and ends in the other.

*Robustness*: a property of an algorithm related to insensitivity to input parameters.

*Scale-dependence*: a property that depends on the scale of a represented model.

*Scanning system* or *scanner*: device for tactile or non-contact acquisition of surface data from a physical object.

*Seed region*: an initial set of data points used for *region growing*.

*Segmentation problem*: the problem to decompose a triangulated surface  $S$  into disjoint connected parts such that their union yields  $S$  and such that each part is of "homogeneous shape" with respect to a specified tolerance  $\tau > 0$  (see Section 4.1).

*Segmenter*: the implementation of a segmentation algorithm on a computer.

*(Segment) extraction*: the process of marking or separating parts of data elements (such as points, edges, or triangles) from an entire data set in order to characterise a common feature.

*Selection*: a mechanism of the natural evolution that allows survival of an individual only if it is adapted to a specific environment (see Section 4.4.1.1).

*Steepest descent*: class of *directional (optimisation) methods* where iteratively the direction of steepest descent in the graph of a function  $f: \mathcal{R}^n \rightarrow \mathcal{R}$  is used to find an optimum.

*Successive segmentation*: segmentation of data such that, once it has been extracted and interpreted in terms of a specific shape, it is unique in type *and* instance.

*Surface classification*: the determination of a set of shapes of which a surface can be composed.

*(Surface) fitting*: the process of adjusting the free parameters  $\mathbf{X} \in \mathcal{R}^m$  of a function  $f: \mathcal{R}^n \times \mathcal{R}^m \rightarrow \mathcal{R}$ ,  $f(\mathbf{P}; \mathbf{X}) = 0$  in order to minimise the sum of values in its dependent parameters  $\mathbf{P} \in \mathcal{R}^n$  with respect to a given mathematical norm; in this thesis the dependent parameters  $\mathbf{P} \in \mathcal{R}^3$  are most often 3D points of a given data set.

*Surface modelling*: producing a representation of surfaces of a 3D scene (i.e. a set of objects) in the real world that can be used for purposes of simulation, analysis, inspection and/or reproduction.

*Surface parametrisation* (in 2 dimensions): the dependency of a surface on parameters such that it can be imposed an order in x- and y-directions when projected appropriately onto a 2D grid.

*(Surface) segment*: a connected set of data elements (of a surface) that logically belong together.

*Surface segmentation*: decomposition of a surface into connected sets of data elements of which each set is homogeneous and unique.

*Termination condition*: a condition causing the termination of a non-deterministic algorithm (roughly an algorithm that solves a problem within a non-predictable number of iterations; see Section 4.4.1.1 for details).

*Triangle-pair method*: generic method of calculating curvature for two adjacent triangles by taking the reciprocal value of the radius of the sphere that passes through the vertices of the triangles.

*Triplet*: three non-collinear 3D points.

*Under-segmentation*: the non-decomposition of a data set such that at least one resulting segment is of inhomogeneous shape.

*Virtual original*: a physical object of presupposed existence as the original from which a given triangulated surface has been produced of and which is approximated by this surface.

## List of symbols

Symbol	Meaning
$ \cdot $	modulus of a real value, also denotes the number of elements in a set
$\ \cdot\ $	length of a vector
$\langle \cdot, \cdot \rangle$	dot product (i.e. standard scalar or inner product) of two vectors
$\times$	symbol for vector multiplication of two vectors
$\alpha$	half of the opening angle of a cone
$\mathbf{a}_{\text{con}}$	axis of a cone
$\mathbf{a}_{\text{cyl}}$	axis of a cylinder
$\mathbf{a}_{\text{tor}}$	main axis of a torus perpendicular to the central symmetry plane
$\mathbf{C}$	general symbol for a centre of a circle or a sphere
$\mathbf{C}^*$	compensated centre of a surface triangle
$\mathbf{C}_{\text{con}}$	apex of a cone
$\mathbf{C}_{\text{cyl}}$	reference point on the axis of a cylinder
$\mathbf{C}_{\text{tor}}$	centroid of a torus (centre of gravity)
$\delta_{\text{max}}$	direction of maximum principal curvature
$\delta_{\text{min}}$	direction of minimum principal curvature
$\partial$	derivative operator
$d$	distance (of a plane or a point from the origin)
$\mathbf{n}$	general symbol for a normal vector (short: normal) on a surface
$\mathbf{n}^*$	compensated normal of a surface triangle
$\mathbf{n}_v$	interpolated normal at a vertex of a triangulated surface
$\mathbf{G}$	set of the geometric primitives plane, sphere, cylinder, cone, and torus
$g$	element of $\mathbf{G}$
$H$	mean curvature at a point on a surface
$\kappa$	curvature value at a point on a curve or surface
$K$	Gaussian curvature at a point on a surface
$\kappa_{\text{max}}$	maximum value of principal curvature
$\kappa_{\text{min}}$	minimum value of principal curvature
$\mathbf{P}$	point in 3D space, i.e. an element of $\mathfrak{R}^3$
$R_{\text{max}}$	principal radius of maximum curvature, i.e. the reciprocal of $\kappa_{\text{max}}$
$R_{\text{min}}$	maximum value of principal curvature, i.e. the reciprocal of $\kappa_{\text{min}}$
$\mathfrak{R}$	set of real values
$\mathfrak{R}^3$	set of 3D points with components in $\mathfrak{R}$
$\mathbf{V}$	vertex of a (surface) triangle
$w$	real-valued weighting factor
$\mathbf{X}$	general vector of characteristic parameters
$\mathbf{X}_g$	vector of characteristic parameters for a specific geometric primitive $g$

# 1 Introduction

In the last few decades the manufacturing of goods and products has become more and more automated. Computers play a key role in this process; they evolved from simple “number crunchers” to tools for highly sophisticated software that supports the design and physical production of objects. An every day object of utility such as a mobile phone casing may be completely designed on a computer and a sample produced by a rapid prototyping facility controlled by another computer.

However, the software in these areas of Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM) is still under development. Some years ago it became possible to create the shape of an object virtually, i.e. on a computer. Shapes generated this way were either relatively simple, or – if they included more complicated details such as a non-standard blend of two adjacent simple shapes – their design became rather time-consuming. Moreover the demand for aesthetically appealing, “smooth” shapes has lead to another technique in digital modelling, namely “reverse engineering”.

So far, complex surfaces have been designed purely using CAD-software, and from the resulting digital description a physical object has been produced (this process is usually understood as “engineering”). On the contrary, the more modern reverse engineering approach takes data obtained from measuring a physical object and aims to create a digital model closely corresponding to the original object. Such a model may enable designers and engineers to improve the quality of the shape before they produce a physical prototype. With this approach the entire production cycle, beginning with the human idea of a visually appealing design for a casing, a tool, a utensil or a vehicle and ending with its effective realisation as a prototype, can be significantly simplified.

The principle of reverse engineering has now been described briefly. In order to track the way from the idea of a shape to its physical realisation, Section 1.1 introduces surface modelling in common applications and points out links to related areas such as object recognition and computer vision. Then the steps of constructing a digital description from a physical object are explained in Section

1.2, namely scanning, registering different views into a single model and generating a digital model. Section 1.3 examines the question why triangulated surfaces are of particular meaning in current CAD systems. Once a digital description is available, Section 1.4 presents motivation for segmenting the corresponding surface into elements of simple shape. The resulting segmentation problem forms the quintessence of the present work. As an appropriate CAD/CAM tool for practical realisations directed towards solutions to this problem, the POint-based MOdelling System POMOS is introduced in Section 1.5. Finally, Section 1.6 outlines the objectives of this project.

## 1.1 Surface modelling

Among many other authors, [Choi 91] refers to **modelling** as the activity of constructing a mathematical or computer model from the description of a shape, usually given in the form of an engineering drawing or stored in a wireframe format. In contrast he employs the term **design** (which is sometimes used as a synonym for modelling) as the activity of creating - in most cases interactively - a geometric shape. On the other hand, he refers to **geometric modelling** as a collection of techniques or tools that may be used in both design and modelling. Various geometric modelling techniques will be presented particularly in Section 1.2.3. However, based on the reference above, within this document the more specific term **surface modelling** will be used as defined next.

*Surface modelling in the present work shall denote producing a representation of surfaces of a 3D scene (i.e. a set of objects) in the real world that can be used for simulation, analysis, inspection and/or reproduction purposes.*

The focus of the present work concerns the analysis and reproduction of a single object in the form of a model surface rather than simulating its physical properties. Furthermore the above definition excludes 2D images as modelling results (such as prints on paper) since they do not provide geometric information for the given purposes, whereas so-called 2½ images (such as a relief) are included. For practical reasons the present work considers mainly 3D models where 2½D models are treated as special cases.

Typical areas of application for surface models are industrial design, medicine and human anatomy, applied arts, fashion, archaeology, and geography. As an example for the application of 3D modelling in applied arts the "Digital Michelangelo Project" described in [Levoy 99] mainly consists in digitising sculptures and statues of Michelangelo for scholarly and educational purposes. As a benefit from this project the 3D reconstruction of statues has offered new insights into Michelangelo's artistic expertise.

A method for 3D modelling and rendering of the human spine, rib cage and pelvis for the study of spinal deformities is presented in [Delorme et al. 99], and a point-based 3D statistical shape model supporting medical image segmentation is introduced in [Lorenz & Krahnstöver 99]. Facial models in three dimensions are reconstructed in [Shihong et al. 99]. [Certain & Stuetzle 99] describe a simple model contributing to the mass customisation of garments. A wider survey of 3D surfaces resulting from body measurements can be found in [Robinette et al. 99], which has been made for the purpose of not only producing better fitting clothes and protective equipment but also better seats and workstations.

For most of the above applications it is desirable to have an accurate digital description of a physical object. However, under certain circumstances it might be too time-consuming to produce a precise image of the object, and a relatively coarse one suffices. Such time-critical modelling tasks are required in computer vision and object recognition. They are related to reverse engineering inasmuch as they employ similar techniques, but they may not aim for the same degree of accuracy, which will be considered next.

Object recognition as a partial discipline in computer vision attempts to explore physical environments by means of sensors in order to identify and distinguish known and unknown objects in terms of size, shape and dimensions. Moreover, the location of objects needs to be inferred from the sampled 3D data (see [Fisher et al. 93]). Such tasks are necessary, for example, when robots or autonomous mobile systems need to perform path planning or collision detection.

On the other hand computer vision aims for the perception of a physical environment in order not only to recognise objects, but also to interpret a 3D

scene, which means, for example, to determine characteristics such as posture, gesture, or features of a person. In every case it is necessary first to create a digital model of a physical object before further analysis can be performed, though the models often do not need to meet the same accuracy standards as in engineering applications.

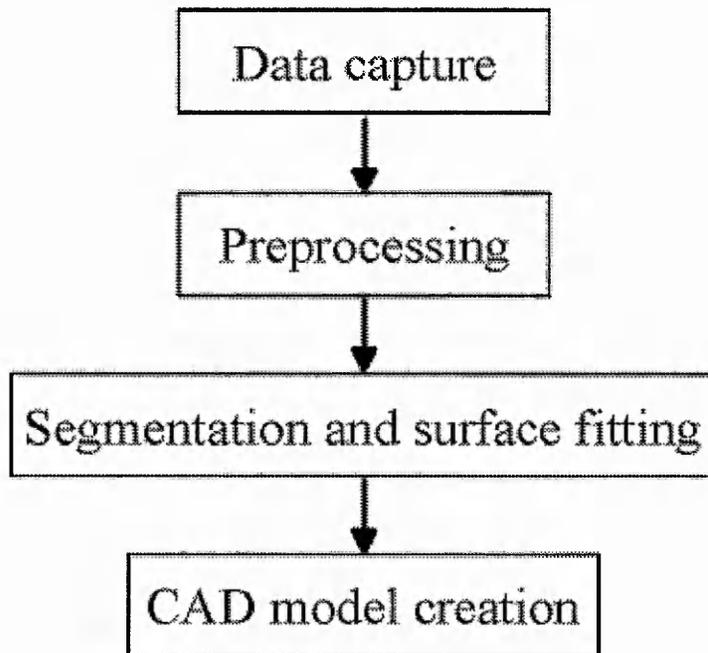
As explained above, surface modelling itself attempts to recreate a digital description from a given physical scene. This may comprise the partitioning of individual objects against the background, the determination of their number, the recovery of their surface shape, shading and texture, finding edge lines, and various other tasks. In order to obtain an appropriate model, a mathematical description must be established first which can then be used for rendering or further digital processing in order to recreate the scene. Section 1.2.3 will present the most common representations used for surface modelling.

In what follows several papers are included on object recognition and computer vision research because many techniques developed in these areas are also beneficial for reverse engineering.

## **1.2 Surface reconstruction by reverse engineering**

Usually the generation of a complete digital 3D model necessitates at least three main steps:

- scanning of a physical object from one or more different viewpoints;
- registering the resulting scans into a single model;
- determining a digital description which fits the entire data set.

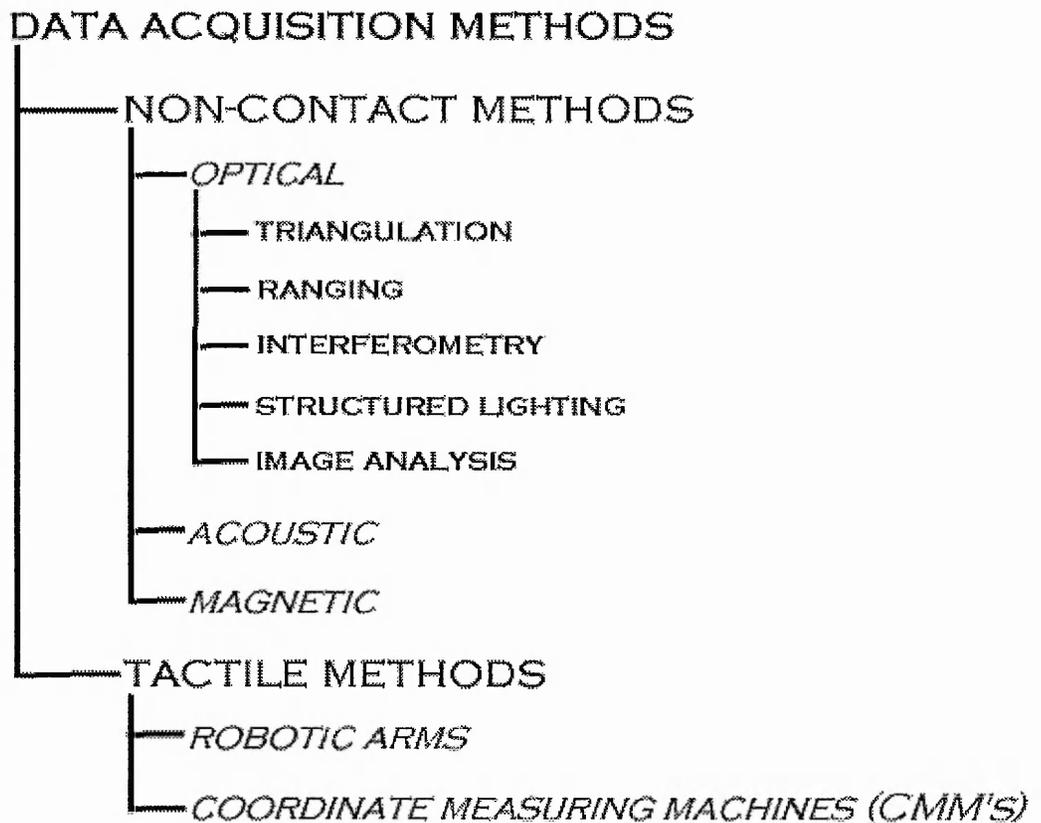


**Figure 1.1:** *Four stages of digital model generation (adapted from [Várady et al. 97])*

Between these other tasks may be required, for example filtering of unreliable data points (“outliers”) or smoothing. This order resembles the scheme presented in [Várady et al. 97] with a difference of further subdivision of the above third stage into segmentation/surface fitting and a separate CAD model generation stage. However, the following sections aim to present a survey of the minimal steps required in order to build a representation of a single physical model in a digitised form.

### **1.2.1 Scanning of a physical object**

Just as 2D photography requires an ordinary camera, so special equipment is required to record 3D surface data from the real world, a so-called **scanning system** or **scanner**. Scanners are generally divided into tactile and contact-less systems according to their method of data sampling (see survey in Figure 1.2).

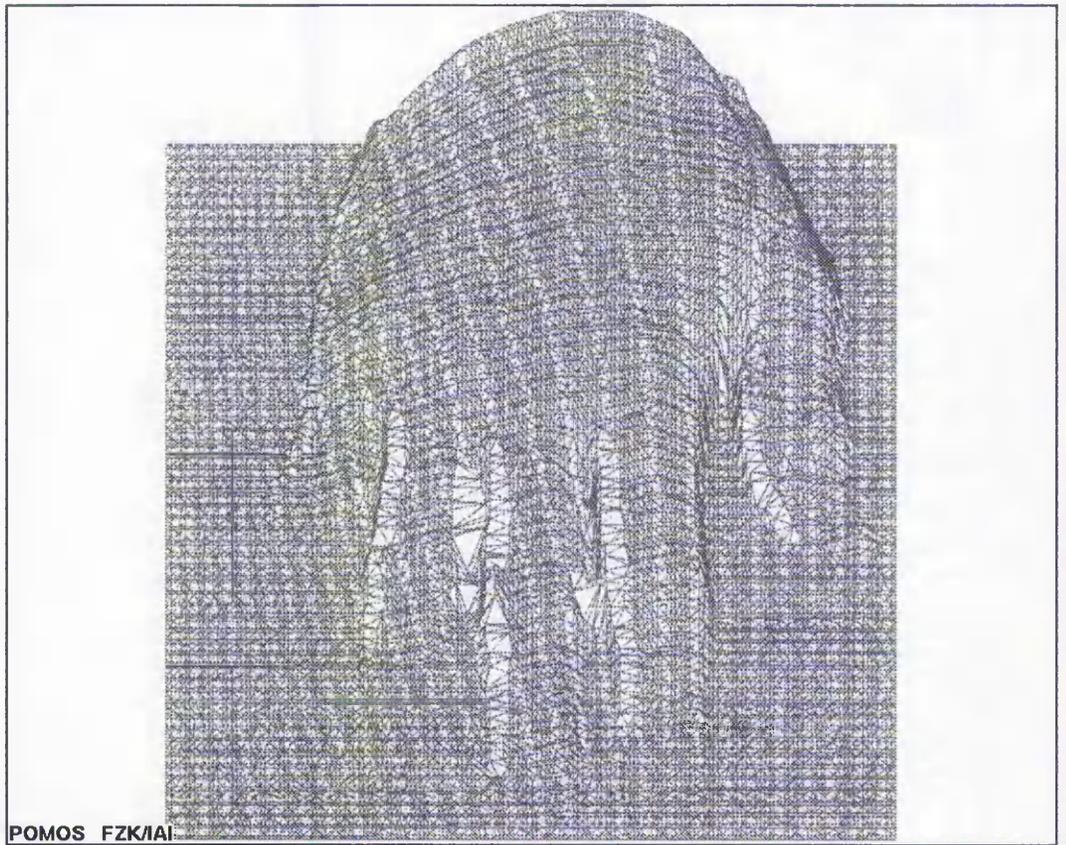


**Figure 1.2:** *Methods of digital data acquisition (adapted from [Várady et al. 97])*

A tactile scanner consists of a multi-axes arm and a touch-probe at its end (the number of axes is usually 3 or 5). The arm is manually or automatically moved along the surface of the object to be scanned. Data points are obtained by determining the position of the touch-probe at certain step intervals or by a manual trigger operation.

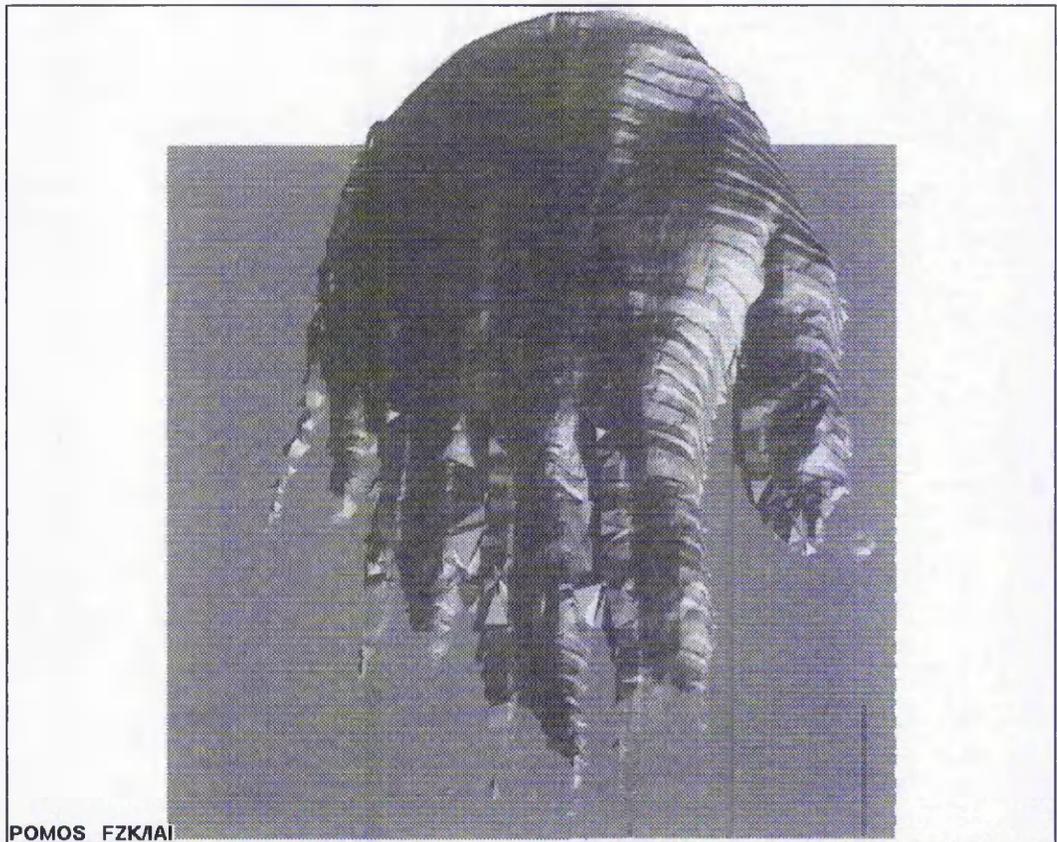
Contact-less scanners can be further categorised into active and passive systems. An active scanner interacts with the surface of the object to be scanned by sending out a signal, whereas a passive scanning system collects signals that are emitted by the object “naturally”, e.g. ambient light. In the latter case such a system may determine surface points by matching landmarks on a surface from different viewpoints, a task requiring further data processing in order to compute a 3D data set of an object. For sampling a data point with an active scanning system a signal is emitted which is reflected by the surface of the object and then captured by a sensor. This allows the determination of the distance between the signal source and the reflection point. Commonly used signal sources are light (e.g. ambient, laser), magnetic fields and sound.

Regardless of the type, each measurement is inherently affected by certain errors, of which some are systematic, others stochastic errors. Systematic errors may arise from inadequate set-up of a scanning system such as errors in calibration, i.e. the determination of optimal values for scanner parameters depending on the object to be scanned. For example, errors may arise from a partly occluded sensor, from a misalignment of the object with the signal source or a sensor, or from an inadequately chosen scaling which leads to false distance measurements. Moreover, during the scanning process an object may not be visible from all viewpoints, if - as is likely - it is attached to other objects such as the ground, a rig, or some cables. This generally results in an incomplete scan that may only be remedied by repeating the scan from a different viewpoint, for which the object may need to change its position. As a consequence scanned data from multiple viewpoints need to be unified in a common coordinate system, a task usually referred to as registration problem (see Section 1.2.2). Some systematic errors occur owing to material properties of the surface to be scanned such as absorption, roughness and transparency. An example of a scanned hand affected by such errors is shown in Figures 1.3 and 1.4. In particular the fingernails in Figure 1.4 have not been sampled properly (possibly owing to reflections).



POMOS FZK/IAI

**Figure 1.3:** *A triangulated mesh of a hand revealing systematic scanning errors around the area of the finger nails*



**Figure 1.4:** *Shaded view of the surface in Figure 1.3 showing artefacts that result from scanning*

Another type of systematic error stems from an insufficient scanning resolution and also produces an incomplete scan. Such an error is likely to neglect smaller features of a physical object and will affect the digital representation later on.

This type of problem can be put in colloquial language as “how to represent an elephant with only 10 data points”. In general the human operator needs to make sure that all relevant features of a physical object can be identified on its digital image. Consequently it will be assumed in the rest of this document that such errors play no role.

Stochastic errors emerge from inaccuracies of the measurement itself when for contact-less scanning the resolution of a signal captured by a sensor is low compared to the distance of an object, or the surface of the object partly disperses a signal away from the sensor. This type of error can lead to, for example, the erroneous determination of landmarks on the object’s surface. Position errors recorded by tactile scanning systems mainly depend on the scanner design, but

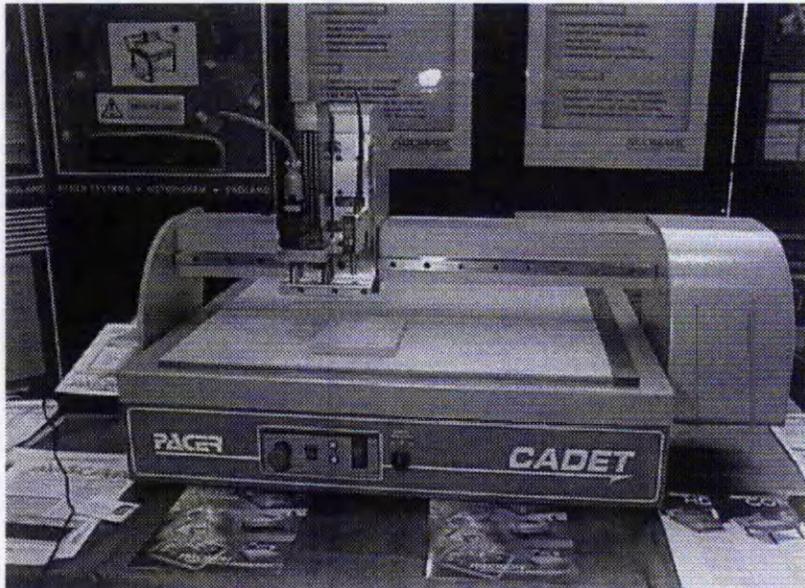
are usually much smaller relative to the object size than those by contact-less measurement. To give an idea about the order of magnitude of position errors in current industrial tactile scanning systems, accuracies about  $150\mu\text{m}$  down to  $1\mu\text{m}$  for mid-size objects (e.g. an appliance casing) may be achieved, when it can be certain that the object does not move.

Finally, some errors may result from “over-sampling” of a surface, i.e. from scanning more than one point within a small radius on an object regardless the type of scanner being employed.

Most desirable for 3D surface scanning would be the system described in [Várady et al. 97]:

*Imagine an ideal scanner: the object is 'floating' in 3D space, so it is accessible from all directions. The data are captured in one coordinate system with high accuracy, with no need for noise filtering and registration. Possibly, the measurement is adaptive, i.e. more points are collected at highly curved surface portions, etc. Unfortunately, such a device does not exist at present. But, despite the practical problems discussed, it is possible to obtain large amounts of surface data in reasonably short periods of time even today using the methods described.*

Examples of contact-less active scanners currently employed (see Figure 1.5) are laser stripe profilers, nuclear magnetic resonance devices (for computer tomography), a so-called “low-cost range finder” ([Fisher et al. 99]), and sonic depth finders (e.g. in [Fusiello et al. 99]), whereas digital cameras and video cameras are regarded as passive scanning systems. The tactile scanners most often used are coordinate measuring machines (CMM's).



**Figure 1.5:** *A typical laser scanning system with a sensor mounted on a portal gantry*

For reverse engineering the scanners most often employed are high-resolution scanners with properly calibrated scanning heads or touch-probes. Since the present work deals with surface modelling for such applications, data sets with relatively small stochastic errors will be presupposed from now on.

Sampled points in such sets are often aligned in scan lines as the result of an automatic line-by-line movement of a scanning head. Each data point is then recorded as a distance in z-direction. This allows the entire scan to be described by a function  $z = f(x, y)$  which has implications on the post-processing of the data (see Section 1.2.3 and Chapter 2). Arising from this description the recorded data set is called a **range image**, since its individual data points reflect the distance from a given reference plane (i.e. a plane defined by an imaginary data set where each point would have distance 0) to the surface or scenery to be scanned. A data set allowing a representation  $z = f(x, y)$  is said to have the **2½D property**, and consequently single range images fulfil this definition.

Characteristic for a pixel in a range image is that adjacent pixels differ in x- or in y-direction only by the distance the scanning head was moved during the data acquisition in each of the directions,  $\Delta x$  and  $\Delta y$ , respectively. Naturally two types of neighbourhood relations are derived from this: a 4-connected pixel at  $(x, y)$  would possess the neighbours  $(x \pm \Delta x, y)$  and  $(x, y \pm \Delta y)$ , an 8-connected pixel would have the same and additionally  $(x \pm \Delta x, y \pm \Delta y)$ .

Now after scans from several viewpoints have been taken, the problem arises how to combine them into a single coordinate frame. The next section addresses this registration problem.

### **1.2.2 Registering multiple views into a single data set**

The previous section discussed how to obtain a partial view from an object resulting in a 2½D data set. However, for modelling purposes designers are mainly interested in rendering and manipulating a complete digital description of an object's surface. Consequently the problem arises how to align data sets obtained as range images from multiple views of an object properly in order to achieve an adequate digital representation. This process of data alignment is here referred to as **registration**.

So far each scan is calibrated relative to the sensor rather than to the object. When a scanner is moved relative to the position of an object in order to prepare the next scan, then this movement is of rigid type that can be mathematically expressed by the appropriate composition of a translation and a rotation operation. Accordingly the scanned data complies with this rigid movement, and therefore - from the sensor's point of view - it is located in a different position. Without additional knowledge about the object, registration can only be carried out when different views overlap sufficiently. A proper registration must satisfy the condition that all points that are nearby on the surface of the object are also nearby in its digital representation after the registration.

Different views of an object may be registered either interactively or automatically. Interactive registration usually implies a higher reliability of the generated digital description, especially if the user brings in prior knowledge. On the other hand, automatic registration of multiple scans is still subject to ongoing research. However, preliminary results indicate its general feasibility.

One class of strategies attempting to overcome the problem of automated registration assumes the availability of a rough initial registration, which allows an iterative refinement via an Iterative Closest Point (ICP) method, mainly

developed by [Besl & McKay 92]. [Pulli 99] explains that “ICP registers two meshes by pairing points in one mesh with nearby points in the other, finding a rigid 3D motion that better aligns the paired mesh locations, and iterating these steps as long as the registration improves.” Based on this he extends the method for multiple scans by an iterative “local matching, global alignment” strategy. Here, local matching involves pairwise matching of points in two different views.

A similar strategy pursues an initial off-line computation of low curvature patches (compare to Section 2.4) prior to an on-line iterative pair-wise matching of points based on the precomputed patches. Thus, this method is denoted by “Robust Closest Patch” (RCP) algorithm ([Nguyen et al. 99]).

The other class of strategies aims to register multiple views without prior knowledge about an initial alignment. [Roth 99] suggests a method where an initial triangulation of each view is based on a set of “interest points”. Thereafter, a matching of all possible triangle pairs is performed, and the best match is determined to be the one aligning the largest number of “interest points”. A more sophisticated approach, likewise applying face-based matching, can be found in [Fischer 99] though it assumes a polyhedral structure of the surface associated with each range image. Here, a similarity measure is constructed from a weighted set of values associated with each facet (such as centroid, area, edge length, etc) that is then employed for a pairwise matching of these facets. The best match determines the alignment of the two views.

Either manually or automatically, a registration of different views can always be achieved. The next important step in surface reconstruction deals with model generation, which is subject of the next section.

### **1.2.3 Model generation**

Customarily, objects are sampled with a resolution according to the specific requirements the digital model needs to satisfy. This can result in data sets as large as a million points. Hence before a surface model is effectively created, data points in range images may require preprocessing such as data filtering (e.g. removal of “outliers” arising from measurement errors or removal of points in an

area of undesirable high point density), "fairing" (correcting the position of data points), "gap filling" (ensuring homogeneous data density), scaling, or trimming (clipping of "sticking out" points at the border). These operations may be carried out on surface patches that have been generated initially to interpolate connected sets of data points. By employing such interpolated patches, properties such as point-to-patch distance, point density, or patch incoherencies can be determined in order to improve the subsequent surface generation.

Generally an object can be modelled by using lines, volumetric elements, or surfaces. Line models represent objects, for example, by edges, contour lines, isophotes, and regular meshes, whereas volumetric models comprise boundary representation (b-rep) and spatial partition (s-part) models. B-rep models describe objects by vertices, edges, and enclosed areas, whereas s-part models mainly combine regular polyhedra to accomplish the same.

Particularly interesting are cubic elements such as a voxel for Volumetric piXEL and octree particles (arising from an initial cube enclosing an object by recursive, but numerically limited, subdivision into sub-cubes wherever the object intersects the cube(s)); cuboids and tetrahedra are less common. Similar to the 2½D adjacency for range images, s-part models also provide spatial connectivity information about their volumetric elements. For rendering purposes usually b-rep models are preferred, for Boolean operations such as intersection and unification (appearing in solid modelling tasks) s-part models are deemed to be more favourable.

There are numerous possibilities for representing a surface, but almost all possess either polyhedral or (higher order) polynomial structure. In some cases combined object representations also make sense when, for example, different signal sources have been merged in order to produce a multi-layer model.

Widely used polyhedral representations include triangular, rectangular, or mixed polygonal facets. Polynomial representations do occur perhaps often in practical applications although the resulting surfaces may interpolate measured data points within a given tolerance rather than fitting them precisely. Examples for polynomial surfaces employ B-spline and Non-Uniform Rational B-Spline

(NURBS) interpolation functions, quadrics and superquadrics, Coons surfaces, parametrised and free-form surfaces. The book of [Hoschek & Lasser 93] offers a good survey of surface modelling in CAD applications.

Among all of the above surface representation types, the present work will particularly set its focus on triangulated surfaces because of their significance in many practical applications; especially because polyhedrons can be easily converted to triangles. Hence the next paragraph will introduce various approaches to triangulate a set of data points in order to generate a triangulated surface.

A simple technique presented in [Häfele & Hellmann 96] can be used in many cases to connect data points of two consecutive scan lines, but it can only be applied to 2½D data. Triangulation of more general 3D data is achieved by [Hoppe 94] as a local-to-global approach in three steps: initial surface estimation (by piecewise linear approximation of spatially adjacent points), mesh optimisation (determining a more concise and accurate mesh exploiting an energy function), and piecewise smooth surface optimisation (determining a new concise tagged mesh defined by a piecewise smooth accurate subdivision surface). [Yemez & Schmitt 99] demonstrate another approach established on octree particles, which further allows a progressive multilevel triangle mesh representation. Here octree surfaces are triangulated by connecting those centroids of adjacent particles that enclose the boundary of an object by edges according to a specific set of rules. Another set of progressive triangle meshes using a “face-centred orthorhombic lattice” is constructed in [Dafas et al. 00]. Beginning with a dense regular mesh an adaptive triangle mesh is created from it by using “fixed position vertices along with an efficient adaptive triangulation technique” followed by a mesh decimation phase. The resulting meshes show more triangles in curved areas and fewer in nearly planar areas.

### **1.3 Significance and examples of triangulated surfaces in CAD systems**

Surface rendering for inspection, analysis, and reproduction usually plays an important role in CAD/CAM systems. A quick method to render a surface by

reverse engineering techniques involves the generation of a triangulation as presented in the last section. As a first step a polyhedral surface is created by connecting adjacent data points as a piecewise linear approximation to a scanned surface. However, every polyhedral facet allows an immediate splitting into triangles by introduction of new edges within each individual polygon, each of which can be rendered almost instantaneously. Moreover, each triangle may be provided with surface colour, texture, and shading in order to enhance the impression of realistic scenery, as often used in computer simulations and games.

Moreover, the interpolation or approximation of triangulated surfaces enables a relatively easy conversion into other surface representations. Some scanners directly provide triangulated data that is favourable in this context. Further applications of triangulated surface data concern numerical methods such as the FEM (Finite Element Method). A typical example involves the computation of heat flow, say, through a continuous surface, which is linearly approximated by a discrete mesh. At each mesh point of the surface the heat flow may then be calculated individually according to the law of thermal conduction. As long as the point distances within the mesh are sufficiently small, FEM fairly reduces the computational overhead and yields in most cases a good approximate solution to the original continuous problem. After the computation the resulting data may be incorporated and processed further by another CAD/CAM tool.

Finally, like the individual data points, triangulated surfaces also allow an interpolation by smooth polynomial surfaces after they have been preprocessed in a way that exploits their triangulated structure. By all these examples it can be understood that the versatility of triangulated surfaces is the reason for their popularity, particularly in reverse engineering applications.

#### **1.4 Motivation for surface segmentation**

Reverse engineering obtains data from objects that are often composed of simple surface shape. For example, the surface of a simple bottle may be composed (from bottom to top) of a circular plane, a cylindrical section, a conic section narrowing upward, and another cylinder of smaller radius on top. In fact, this just

reflects the underlying engineering principles where often a fairly small set of primitive shapes forms the basis of designing more complicated shapes.

It is therefore understandable why a user of a CAD/CAM systems employing reverse engineering is commonly interested not only in the shape of a surface itself, but also in gaining more information about its components. In particular, engineers would like to have a suitable surface segmentation into parts of appropriate geometric primitives. Such segmentation, when provided with corresponding shape parameters, may speed up subsequent manual design modifications as well as rapid prototyping for a physical object reproduction.

While designing a surface interactively on a computer screen by means of a CAD/CAM system, a human user may wish to include "real world" data to combine both, synthetic and natural surfaces within one model. For the extraction, removal, substitution, or attachment of surface parts a segmentation of either of the surfaces is likely to be necessary. Moreover, the shape parameters may control rapid prototyping facilities for a precise digital-to-physical model reconstruction.

Furthermore, segmentation is an essential task required in computer vision and object recognition. Segments of geometric primitives such as planes, spheres, cylinders, cones, or tori can often be easily detected by a human being (although it may not always be clear whether a small portion of a curved surface looks more like part of a sphere or a torus, say), but it is still a challenge for a computer to achieve the same without human intervention. At first, within a given scene the ground needs to be separated from non-ground objects, and then each individual object considered as a "sensible component" needs to be identified. For this identification, shape characteristics can be used once they have been determined. Such characteristics in turn may be obtained from a component analysis of the segmented surface of the object.

Moreover, surface segmentation may form a basis for the generation of a high level CAD model hierarchy, in which objects are classified according to their number of segments, their shape parameters, and their boundaries. Such a

hierarchy may be provided with Boolean operations in order to allow comparison, union, and intersection operations of objects.

Finally, a useful side effect of surface segmentation concerns data reduction. For objects given as high level models consisting of only a few shapes, characteristic parameter values and boundary curve values obtained from a surface segmentation data transmission will be more efficient than transmission of a large number of surface coordinates.

### **1.5 POMOS - the implementation framework**

As only very few of the current CAD/CAM systems can perform operations on huge point clouds, surface generation, and surface analysis at the same time, most such systems are still part of ongoing research projects. One of these was recently developed at the Research Centre Karlsruhe in Germany and is designed to close the gap between measuring devices and subsequent CAD/CAM applications ([Häfele 96]), and thus it is an appropriate testing platform for the present project. Its name POMOS stands for POINT-based MODelling System. Besides data acquisition, surface generation, and graphical file conversion capabilities it also offers the benefit of processing triangulated surfaces from point clouds, whereby it is distinguished as a pertinent framework for the investigations made in this project. Particularly, input 2½D point data can be triangulated automatically, and the resulting triangles are provided with consistently oriented normal vectors.

Thus the data flow in POMOS can be schematised as follows:

1. Data input (points, curves, surfaces, facets)
2. Data processing (sorting, splitting, smoothing)
3. Geometry generation (approximation of curves and surfaces)
4. Geometry analysis (normal vectors, cutting lines, isophotes)
5. Data output (points, surfaces, isolines, etc.)

Steps 2 - 4 may be processed iteratively until the result visually satisfies the demands of the human designer.

Some of the above tasks realised in POMOS involve user interaction such as the splitting of data or the selection of approximation curves and surfaces with their corresponding shape parameters. However, the splitting and the geometry generation may be unified by approximating a subset of data points with appropriate curves and surfaces in a manner which requires almost no user interaction. A more precise description of this task will be presented in the next section addressing the objectives of the present work.

## **1.6 Objectives of this project**

The overall aim of this project is to develop strategies for automatic segmentation of triangulated surfaces into components of homogeneous surface shape. It is assumed that the triangles are provided with connectivity information (allowing direct access to adjacent triangles from a given triangle) as well as with consistently oriented surface normals of unit length. Furthermore, it makes sense to demand that the minimum length of all edges on the triangulated surface (i.e. the smallest distance between two sampled data points) is “sufficiently” large compared to the maximum error resulting from the measuring process in order to avoid “over-sampling”. No assumptions are required for the mesh being open or closed.

In general the segmentation problem has no unique solution as, for example, a cone shell may be either approximated by a sequence of narrow triangles, by a sequence of cylinder shells of low height, or by another cone with a minor change of its opening angle. Hence the current project aims for a practical solution. For example, the segment of a geometric primitive of lowest order (ascending: plane, sphere, cylinder, cone, torus) may be considered as the best local approximation of the triangulated surface provided that it has a sufficiently large area in comparison to the total surface area.

Thus, effectively for each input polyhedral surface, regardless of its convexity or concavity, a b-rep model consisting of a list of segments of geometric primitives shall be output. Moreover, for each segment the corresponding boundary curves and its characteristic parameters (i.e. a set of parameters that uniquely describe a specific instance of a geometric primitive) shall also be produced.

In order to obtain such segmentation it will be necessary to:

- a. Define a precise problem specification;
- b. Establish an initial surface shape classification method for triangulated surfaces by introducing curvature estimation;
- c. Develop methods of surface patch identification for homogeneous geometric shapes followed by separation from the remaining data (surface fitting and feature extraction);
- d. Establish methods for the estimation of characteristic parameters for each type of geometric primitive to be identified;
- e. Evaluate segmentation results including a comparison of different strategies.

Since the idea of decomposing a triangulated surface into parts of homogeneous shape is evident in theory but so far has not been realised in practice, the primary objective of this work is to offer a practical solution to the segmentation problem. For this particularly the following objectives need to be achieved:

1. A definition of the segmentation problem appropriate to the unstructured triangulated data;
2. A mathematical surface representation of geometric primitives reflecting point-to-surface distances for numerical processing (such as surface fitting);
3. Appropriate algorithms for curvature estimation for unstructured triangulated data;
4. Algorithms for the identification and extraction of segments of geometric primitives;
5. Algorithms for accurate estimation of characteristic parameters for segments of geometric primitives;
6. An appropriate numerical optimisation method for iterative characteristic parameter readjustment;
7. Implementation of methods and algorithms in 3 - 6 into a suitable framework for evaluation (POMOS);
8. The evaluation of segmentation results including a comparison of some selected segmentation strategies.

## 2 Survey of surface segmentation

A surface may be decomposed with respect to various criteria, e.g. into segments of identical area or volume (the latter can be accomplished only with respect to the convex hull of the surface), homogeneous shapes, similar colour, texture, or other physical properties. However, for applications in computer-aided geometric design a decomposition into “smoothly” connected components of simple shape is most desirable. Simple shapes usually are mathematically represented by an explicit function such as a polynomial (e.g. B-spline, NURBS, quadric, superquadric), or by an implicit function. Because the representation affects the types of possible shapes that are used to approximate a digital surface, it also affects the result of a **segmenter** (i.e. the implementation of a segmentation algorithm on a computer). For example, the segmentation result for a representation only allowing planes will generally differ from one that only allows spheres. Furthermore, the measured distance from a point to a part of an underlying surface that is to be extracted also depends on the representation.

Each segmenter needs to have some “inbuilt notion” about the properties of the segmentation result independently from the input data. One part of this notion concerns the digital surface representation, the other an appropriate specification of the segmentation problem. So an abstract problem definition may help the reader to understand the requirements for a “good result” obtained from a segmenter.

The decomposition of a digitised surface into smooth surface components can be achieved in various ways. One class of “top-down” approaches attempts to detect such components within the image by looking for local “surface discontinuities” based on point distances. In a second step the individual components are then merged to larger connected components. A class of “bottom-up” approaches places a small set of seed points into the image and then attempts to “grow” this region by adding adjacent points, as long as they represent a surface of simple shape within a certain tolerance. The common feature of another class of segmentation approaches consists of grouping data points according to surface-based properties or point distances. Furthermore, combinations of these approaches as well as special techniques are used. In support of the search for smooth surfaces, local surface properties such as curvature may be estimated.

Moreover, curvature belongs to the surface-based properties that can guide a surface segmentation process.

Typically, existing approaches and techniques for surface segmentation are applicable to range images only. This is because often such images arise “naturally” in the first step of digital surface modelling, namely after a surface has been scanned from a single viewpoint. These techniques most often rely on a **surface parametrisation** (i.e. the order of data points in x- and y-directions when projected appropriately onto a 2D grid), which is in general not available for “real” 3D data. This may be the reason why the segmentation of triangulated surfaces so far has received little attention. However, the application of a bottom-up approach to this type of surfaces has shown to be applicable ([Fisher et al. 97]).

The remainder of the chapter is organised as follows. Firstly, suitable mathematical surface representations are introduced in Section 2.1. Next, Section 2.2 offers a survey of previous problem specifications as well as segmentation approaches (top-down, bottom-up) and techniques (clustering, special techniques) which have been used for specific surface representations. The role of surface curvature and its application for the determination of parts of a specific surface shape is explained in Section 2.3. Thereafter “region growing” as the method of choice for this project is presented on a more detailed level in Section 2.4 involving an estimate of characteristic parameters of a surface, surface fitting, and segment extraction. The summary in Section 2.5 discusses the previous work and addresses the open problems this project aims to solve. Finally, Section 2.6 explains the constraints and techniques that are relevant to this project.

## **2.1 Surface representation and classification**

As described in the first chapter a digital model can be represented in many ways, such as lines, volumetric elements, implicit functions, or surfaces. Since each volumetric element model can be converted to a surface model (but not vice versa because, for example, a plane can at best be represented as a flat box) only surface models will be considered from now on.

In the field of CAD numerous surface representations are known. For example, [Choi 91] lists five ways of describing surfaces on engineering drawings:

- *by the use of surface primitives;*
- *as a mesh of curves;*
- *as a sweeping of cross section curves;*
- *as a set of 3D points;*
- *as a blending of two or more surfaces.*

Furthermore he explains the term “surface primitives” as follows:

*“A simple method of defining a surface is the use of surface primitives, for example quadric surfaces, which can be exactly specified by a few parameters. ... In order to define a useful surface, however, individual surface primitives may need to be ‘trimmed’ and/or ‘compounded’ (meaning that the entire surface is a Boolean sum of individual primitives).”*

With respect to 1.6 the objective of surface segmentation is to achieve the task that is the opposite of “compounding” a surface. That is to assume it is compounded as a Boolean sum of individual geometric primitives and to determine an appropriate decomposition into segments of geometric primitive (or shape) types. The decision which types are used for the segmentation of a given set of data points or a digital surface representation is here denoted by **surface classification**. A standard set of shape types for classification comprise plane, sphere, cylinder, cone, and torus. Such a set is commonly employed in engineering applications as well as being subject to ongoing research in surface segmentation.

Since the aim is to process scanned data automatically, the focus will be on representations which are of particular interest for reverse engineering. This implies that surface types such as, for example, general paraboloids of revolution, will play no role here although they may have some applications in CAD tools. Representations arising from the interpolation of scan lines or involving data points with adjacency information are of main interest instead.

An important property of surfaces is the ability to allow a parametrisation, where a parametric surface is the image of a regular mapping  $f(u,v)$  of a set of points in a 2D domain  $\Delta$  into  $\mathbb{R}^3$  ([Choi 91]). The mapping function

$$f(u,v) = (x(u, v), y(u, v), z(u, v)) \quad (2.1)$$

is called the **parametric equation of the surface**. When the domain  $\Delta$  is defined on the  $x,y$ -plane of a Cartesian coordinate system, the above surface equation can be simplified to the form

$$z = g(x, y). \quad (2.2)$$

The above expression, introduced in 1.2.1, is an **explicit representation** of a surface  $S$  and only suitable for  $2\frac{1}{2}D$  data.

Having a parametrisation of a surface is often essential since many of the approaches and techniques are based on this property. It provides an immediate determination of directly connected pixels (i.e. data points) in range images. Moreover, a globally parametrised surface forces such an image to be topologically equivalent to a plane. The adjacency degree (i.e. the number of adjacent pixels) associated with each interior pixel in a range image is either 4 or 8 depending on the neighbourhood relationship considered (compare to end of Section 1.2.1).

For what follows it is convenient to introduce the **implicit equation**

$$f(x, y, z) = 0 \quad (2.3)$$

for a mapping  $f: \mathfrak{R}^3 \rightarrow \mathfrak{R}$ , which may define a surface in a 3D domain  $\Delta$ . For example, a sphere with centre in the coordinate origin and radius  $r$  is represented by the implicit equation

$$\|(x, y, z)\| - r = 0, \quad (2.4)$$

where  $\| \cdot \|$  denotes the Euclidean norm in  $\mathfrak{R}^3$ . By defining

$$f(x, y, z) = z - g(x, y) \quad (2.5)$$

it is clear that each explicit representation of a surface can be converted to the implicit form, but not always vice versa (consider e.g. the above implicit equation of a sphere). This proves that more surfaces can be represented by implicit rather than explicit equations.

When a data point  $\mathbf{P} = (x, y, z)$  lies exactly on a digital surface, the implicit function  $f$  satisfies

$$f(\mathbf{P}; \mathbf{X}) = 0. \quad (2.6)$$

for an appropriate vector  $\mathbf{X}$  of characteristic parameters which describe the shape. So for a given vector  $\mathbf{X}$  this representation could be used to generate a digital surface, for example, by determining points  $\mathbf{P}_i$  that satisfy the above equation (although a parametric representation simplifies this task). However, in order to obtain a segmentation of such a surface just the opposite problem needs to be solved: determine a vector  $\mathbf{X}$  of characteristic parameters for a given subset  $\Omega$  of data points and for a given function  $f$  used to approximate a digital surface such that the overall distance, modelled by a function  $d$ , of the points in  $\Omega$  from the surface represented by  $f$  is “as small as possible”. Most often this **goodness of fit** is expressed in terms of least-squares of the overall distance  $d$  given by

$$d(\Omega, \mathbf{X}) = \sum_{\mathbf{P} \in \Omega} (f(\mathbf{P}; \mathbf{X}))^2 \quad (2.7)$$

or, less commonly, in terms of the maximum error

$$d(\Omega, \mathbf{X}) = \max_{\mathbf{P} \in \Omega} |f(\mathbf{P}; \mathbf{X})|. \quad (2.8)$$

Thus an essential part of the segmentation problem consists of the minimisation of  $d(\Omega, \mathbf{X})$  with respect to  $\mathbf{X}$ .

NB: None of the above equations is unique since each non-zero real-valued multiple  $\lambda d(\Omega, \mathbf{X})$  has the same set of zeros as  $d(\Omega, \mathbf{X})$ . However, each equation is unique in reflecting the real Euclidean distance between a data point and the surface of the geometric primitive with respect to its given characteristic parameters. The linearisation of these equations yields only first-order distance approximations (such as in Section IV of [Taubin 91] or in [Lukács et al. 98]) but may make a subsequent optimisation of the characteristic parameters easier.

## 2.2 Previous work in surface segmentation for range images

In order to obtain a “high-level” surface description, it is necessary to analyse and group the present geometric data into sets  $\Omega_i$  to each of which an appropriate single surface can be fitted. Appropriate surface types are those specified previously within the shape classification. The **fitting** itself may be performed in many different ways, though it is common to choose the shape which fits “best” in the sense of approximating as many data points as possible.

A dilemma results from the fact that no a priori information about the characteristic parameters fitting to the data is known. [Besl & Jain 88] consider it as a "chicken and egg" problem: if the characteristic parameters of a surface were given, then the set of data points fitting these parameters could be determined. Or, vice versa, if a subset  $\Omega$  of data points to be fitted were given, its characteristic parameters might be obtained from this. In fact, both point subsets  $\Omega_i$  and the corresponding vector  $\mathbf{X}$  of characteristic parameters need to be determined simultaneously. Previous work in this area is based on range data and presented from Section 2.2.1 onwards till the end of the chapter unless otherwise stated.

Possibly this dilemma can be solved by allowing a suitable tolerance for a point subset to differ from the surface to be fitted. Such a tolerance parameter can be useful to compensate for data that is affected by measurement errors, henceforth denoted as **noisy data**. Once such a parameter has been preset, then that shape type (with corresponding characteristic parameters) is considered as the best fit to the data that covers a maximum number of points. Despite this pragmatic solution, ambiguities may still occur. For example, for a given tolerance the same subset of points may be interpreted either as of planar, spherical or other type, where the approximating sphere (or cylinder, or torus) possesses a sufficiently large radius. [Besl & Jain 88] suggest that the simplest surface type (i.e. such with the least number of parameters) may be preferred in such cases.

Another dilemma arises whether to accept a relatively small number of covered data points with a good fit, or whether to prefer a slightly worse fit with higher number of interpolated data points instead. The latter dilemma is equivalent to the problem how to choose the value for the tolerance properly. If a tolerance is given, then a shape and its characteristic parameters fitting to the maximum number of data points may be determined from this. On the contrary, given a shape with its characteristic parameters and a subset of points fitting to these, the appropriate tolerance can then be calculated. An approach for automatic derivation of such a tolerance may be found, e.g. in [Hilton et al. 95] where it is called "threshold". However, since it is in general difficult to determine such a tolerance, for the present its existence is presupposed as interactively defined by a human expert. Section 5.2.1 offers a method of how a proper tolerance might be determined from the data automatically.

Assuming that the surface fitting was successful, the identified shape segments need to be marked or separated from the remaining data in order to gain useful segmentation results. This separation process is referred to as (**segment**) **extraction**. It can be omitted if a reasonable grouping of the data has already been achieved beforehand such that only surface fitting of data subsets would be required.

### 2.2.1 The segmentation problem

[Hoover et al. 96] explain that “*informally, segmenting a range image is the process of labeling the pixels so that pixels whose measurements are of the same surface type are given the same label*”.

Though the formal definitions of the segmentation problem in the existing literature differ slightly, the following specification given by [Gonzalez & Woods 92] can be considered as typical.

Let  $R$  represent the entire image region. Then segmentation can be viewed as a process that for some  $n > 0$  decomposes  $R$  into  $n$  disjoint subregions  $R_1, \dots, R_n$  such that

- their union yields  $R$ ;
- each  $R_i$  is a connected region;
- $P(R_i) = \text{TRUE}$  for  $i = 1, \dots, n$ ;
- $P(R_i \cup R_j) = \text{FALSE}$  for  $i \neq j$ ,

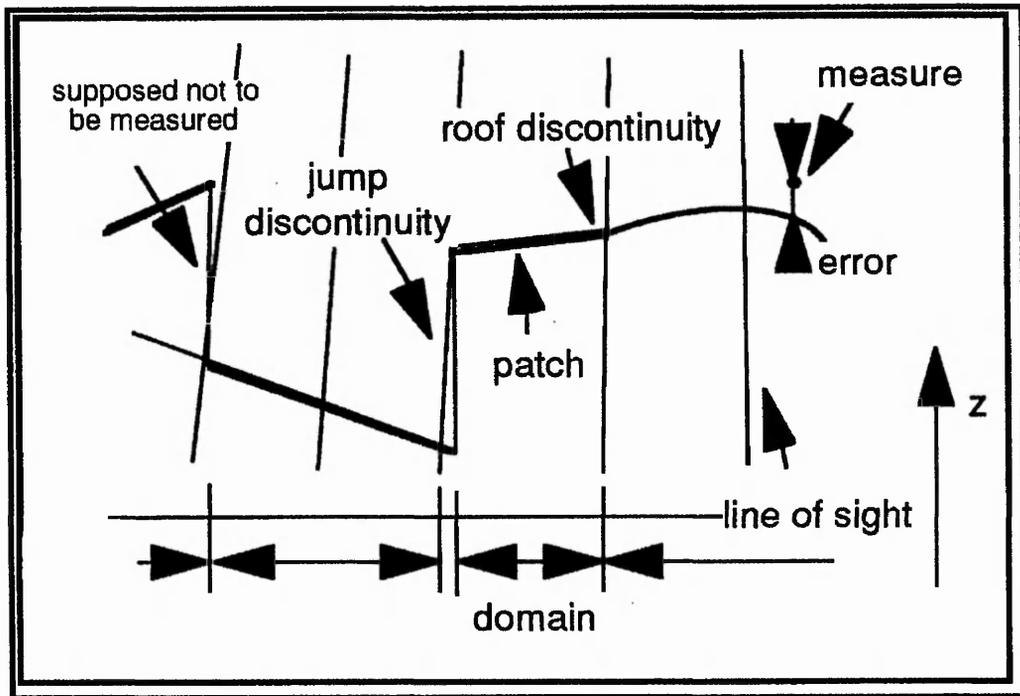
where  $P(S)$  is a logical predicate characterising the homogeneity of a surface shape over points in a subset  $S \subseteq R$ .

More formal definitions for the segmentation of range images can be found, for example, in [Ballard & Brown 82], [Maître et al. 90], and [Leonardis et al. 95], which differ slightly from each other. The last authors refer to [Horowitz & Pavlidis 74] and [Zucker 76] for the definition of the classical segmentation problem. Unlike the others, [Ballard & Brown 82] and [Leonardis et al. 95] permit overlapping segments. On the other hand [Gonzalez & Woods 92] as well as [Maître et al. 90] demand that no two segments have identical shape and characteristic parameters. This strict condition is relaxed in [Hoover et al. 96], where it is postulated only for adjacent segments, since non-bordering segments

are allowed to have the same shape and characteristic parameters (e.g. when a cylinder is dissipated into two segments by a flat intersecting cuboid). As another difference between the papers, the problem specification in [Leonardis et al. 95] allows non-complete decompositions, so that the union of the subregions may be a subset of the image region only rather than the entire region.

The expression “homogeneity of a surface shape” in the segmentation problem definition above requires further explanation. It may be expressed, for example, in terms of convexity, concavity, planarity, in terms of possessing a polynomial surface description of increasing order (such as quadrics/superquadrics), or by the shape of geometric primitives. Previous segmentation approaches include decompositions of a range image into smooth components with spatial discontinuities in between. For example, [Maître et al. 90] enumerate three types of discontinuity: “*measurement gap*”, “*jump discontinuity*” and “*roof discontinuity*” (see Figure 2.2.1). [Chen & Liu 97] additionally list discontinuities in the second derivatives of adjacent surfaces approximating the data.

Although the above conditions for the segmentation problem are desirable in theory, they are scarcely grantable in practice. Segmentation processes can perform **over-segmentation**, where a homogeneous surface part is split into multiple instances of the same or of a different shape type, resulting “*in an incorrect topology*” ([Hoover et al. 96]). Another possible unwanted result occurs in **under-segmentation**, where a non-homogeneous surface part is not decomposed into smaller segments of homogeneous shape resulting “*in a subset of the correct topology and a deformed geometry*” (ditto). Furthermore, for some surface parts no shape may be extracted, thus the corresponding segment needs to be classified as a **missed segment**. Finally, a segment is characterised as a **noise segment** if the segmentation process finds a non-existent shape in the data.



**Figure 2.2.1:** *Decomposition of a sensed surface using surface discontinuities (taken from [Maitre et al. 90])*

Currently proposed or developed segmenters also differ in the order surface fitting is performed. Some of them segment **concurrently**, which means they fit different shapes to the same data subset which may cause (at least temporary) ambiguities in the interpretation of the data. Other segmenters operate **successively** such that data, once it is extracted and interpreted in terms of a specific shape, is not available for another interpretation.

Both approaches offer problems and opportunities. Successive segmentation can leave the order of shapes to be fitted and extracted to the user, but when the shapes can be clearly distinguished this may be acceptable. In contrast to this, concurrent segmentation is especially designed to automate the selection of which shapes meet the “best fit” condition. However, problems can arise from overlapping regions where it may be ambiguous to which segment (defined by a specific shape, a corresponding vector of characteristic parameters, and a boundary curve) parts of a surface belong to. Thus it may happen that the evaluation of a distance function assigns a noisy data point to *one* segment, while it is entirely surrounded by less noisy points that are assigned to *another* segment. Moreover, the boundary between two adjacent segments created in this way is likely to emerge considerably jagged. Hence this approach, although more promising than its successive counterpart, requires careful implementation.

Most effort has been spent on developing segmentation techniques for range images possessing the 2½D data property (see e.g. [Chen & Liu 97] or [Veelaert 97]). As many genuine reverse engineering applications offer “real” 3D data, many such segmentation techniques for range images cannot be applied directly. However, in the next section some segmentation techniques applied to range images are presented, and specific problems in extending these to techniques to 3D data are pointed out.

### **2.2.2 Top-down approaches**

A common feature of top-down approaches employed for surface segmentation involves a recursive refinement of hypothesis generation and validation. First the hypothesis is made that all data points belong to a single surface segment, and then this hypothesis is tested for validity, where the hypothesis depends on a tolerance criterion such as introduced above. If the points are consistent, the recursion terminates; otherwise the points are subdivided into several subsets, and thereafter the single-surface hypothesis is applied recursively to these subsets. The recursion continues until all generated subsets satisfy an individually adequate hypothesis.

The recursive subdivision is usually referred to as “split phase” as the surface is split up into consistent subsets. After the split phase two or more adjacent subsets could satisfy the same hypothesis although they are not joined. Therefore in an iterative “merge phase” these subsets are fused to larger sets of data points. The merge phase is terminated when no pair of adjacent subsets can be fused without violating their corresponding tolerance-based hypotheses. Accordingly algorithms following this two-phase scheme are classified as split-and-merge algorithms.

Although top-down approaches are less popular than the bottom-up ones presented in 2.2.3, they have some relevance when planar data needs to be extracted from range images. Typical examples may be found in [Taylor et al. 89], [Jiang & Bunke 94] and [Fischer et al. 99], where split-and-merge techniques are mainly employed for the fitting and extraction of planar segments.

The above approach brings with it the problem of determining an appropriate subdivision of a data set. For example, when considering a parametric range image with a crease along the x-y-diagonal, a subdivision based on either x- or y-axis is likely to produce rather fragmented data subsets, unless the surface geometry is taken into account, which is yet a priori unknown. Thus, after merging fragments of same shape, the resulting boundary curves of segments may appear fairly jagged.

The most grave restriction of the split-and-merge approach is its requirement for parametrised data, i.e. data that can be aligned on a regular grid in x- and y-coordinates. In general a triangulated surface cannot be parametrised as it allows for each data point a variable number of neighbours (consider, for example, the apex of a cone, where all adjacent vertices lie on the same base plane). Moreover, such a surface may topologically be equivalent to a surface with a number of “handles” (each of which is topologically equivalent to a torus) so that it cannot be mapped globally onto a sphere. The latter property exacerbates the finding of appropriate splitting points. For this reason nothing has been found in the existing literature about the applications of split-and-merge algorithms to non-parametrised data.

### **2.2.3 Bottom-up approaches**

Bottom-up approaches are distinguished by an iterative hypothesis refinement and validation scheme based on a good initial guess. Initially a hypothesis about shape type and characteristic parameters is deduced from a small set of data points, and this will be validated - and if necessary adapted - every time after new points have been added to the set. Iteration terminates when all unused points that could be included violate the hypothesis and an adaptation is impossible. Then the entire process will be restarted with another initial set of points until all data points satisfy an adequate hypothesis.

The initial set of data points mentioned above is often denoted as “seed points” or **seed region**, and the process of successive addition of new points to the seed is sometimes referred to as “seed expansion” (e.g. in [Maître et al. 90]), but more often as **region growing** ([Faugeras et al. 83], [Besl & Jain 88], [Taubin 91], [Leonardis et al. 95], etc.) as throughout this document. Deduction of

characteristic parameters from a seed region will be called **characteristic parameter estimation**, and their iterative adaptation during the region growing phase **parameter optimisation**. This approach attracted much attention based on the work of [Besl & Jain 88], and further refinements as well as variations may be found, for example, in [Taubin 91], [Roth & Levine 93], [Leonardis et al. 95] and [Veelaert 97]. A basic region growing algorithm is presented in Section 2.4.

Various problems originate from the above approach. First of all, it is unclear how to select appropriate seed points, since a random choice may affect the initial hypothesis in both, shape type and characteristic parameters, unfavourably, and this will affect the entire region growing process. Another open question is how to deduce primitive type and corresponding characteristic parameters from a seed region. Once these parameters have been estimated, the problem arises how to adjust them to new points that are attached to the seed region after a growing step. Moreover, how shall this adjustment cope with “outliers” in the data? During region growing it may emerge that - owing to noisy data - a region initialised as a sphere is indeed of cylindrical, conical, or toroidal shape. Thus where necessary it must be decided either to keep the point set as a small consistent surface segment, or to change the surface type.

However, the above problems are of general nature, and the particular structure of triangulated surfaces does not cause further problems. An advantage is that after the individual problems associated with the above approach have been solved, region growing can be processed in linear time, and an algorithm employing this bottom-up approach enables a concurrent (i.e. a parallel) implementation. So [Leonardis et al. 95] summarise:

*“The key idea to independently and redundantly recover surface models makes the scheme fully parallelizable and thus suitable for implementation on parallel computer architectures”.*

## **2.2.4 Clustering techniques**

Until now different clustering techniques in range and image data segmentation have been field-tested, as they are quite robust against processing of noisy data points. A characteristic of all such techniques is the grouping of the data according to surface-based properties or point distances. Examples employing surface-based properties comprise the histogram approaches in [Han et al. 87] and

[Biswas et al. 95], the Hough transform used in [Hebert & Ponce 82], and the WSU (Washington State University) range segmentation algorithm in [Hoover et al. 96]. Point distances are used in the random sampling method of [Roth & Levine 93] or for the residual consensus approach in [Yu et al. 94].

The WSU range segmenter in [Hoover et al. 96] feeds each data point together with an estimated surface normal as a six-dimensional vector into a squared error clustering algorithm. Subsequent processing encompasses a cluster-to-image conversion, an edge-based “*domain independent*” merging of adjacent segments, and an iterative phase of segment classification, merging of adjacent segments of identical type, and merging of pixels at segment boundaries. In [Biswas et al. 95] the clustering is based on a surface orientation described by a 13-part set of “digital neighbourhood planes”. Methods presented in [Roth & Levine 93] and [Yu et al. 94] employ stochastic data point selection as a method of surface sampling in order to determine connected surface components, whereby points with similar geometric distances (residuals) from an assumed underlying surface of a certain geometrical type are likely to be assigned to the same cluster. [Ng et al. 95] employ clustering on characteristic parameters that have been deduced from a pair of local surface patches and then accumulated.

Because of the varying approaches, existing problems of these techniques are hard to generalise. The methods described in [Biswas et al. 95] and [Hoover et al. 96] suffer mainly from the drawback that they are only useful for a coarse segmentation. In particular the first of the two methods is designed only to extract planar segments. Although these problems do not exist for the stochastic techniques, they are computationally intensive, so it is recommended to speed them up by parallel computing.

Again specific difficulties arise in applying surface-based clustering directly to triangulated surfaces, since there is no parameterisation. On the other hand the two latter stochastic methods cause problems because the average number of data points in triangulated surfaces is higher than in range or image data sets. However, a common feature of all the above approaches is that they generally output non-connected regions which would require further postprocessing for the determination of connected data.

### 2.2.5 Special techniques and additional remarks

A “natural” segmentation approach would involve splitting a triangulated surface along highly curved regions (also called “edges”) that arise from non-smooth joins of otherwise smooth surface parts. Such an approach would require the detection of splitting curves. Additionally [Wilke 94] aims to detect curves along which the curvature of a triangulated surface changes in sign (see next section for details). However, this approach fails either when the surface has no detectable “edges” or when such an “edge” ends in a flatter region. Moreover, **edge-detection** turns out not to be sufficiently robust to spurious data points. An improved method is presented in [Hoschek et al. 98], where an attempt is made to determine “feature lines” deduced from the angular variation of estimated normal vectors. Surface segments originating from the separation by such feature lines are locally approximated by B-spline surfaces in order to improve their quality.

One might argue that the Hough transform could be an appropriate tool for the detection of homogeneous shape. The Hough transform maps features from a (not necessarily range) image such as 2D “edgels” (short edge elements in the direction of the tangent of a curve) into a discretised parameter space, the so-called Hough space. Each feature “votes” for its corresponding parameter vector in the Hough space such that features occurring most often result in peaks. In fact, this technique can be generalised to higher dimensions as discussed, for example, in [Wright et al. 96]. However, when the discretisation of the parameter space is not very fine, the determined parameters are not very accurate. On the other hand if the discretisation is fine, then the required computational memory - especially in multiple dimensions - exceeds practicable limits. For these reasons other segmentation techniques seem to be more favourable.

For the localisation of planar segments in range images special techniques have been developed such as in [Taylor et al. 89] or [Jiang & Bunke 94]. A variation of the split-and-merge approach can be found in [Taylor et al. 89], where in a preprocessing step for each point a least-squares plane is estimated in small pixel neighbourhood whose parameters are then converted into spherical coordinates. So each data point is associated with this plane that potentially includes the point. Split-and-merge is then applied within the parameter space (describing planes in spherical coordinates) instead of the data level. This approach bears similarities

to the Hough transform described above with very similar problems for large data sets. Moreover it is restricted to range data.

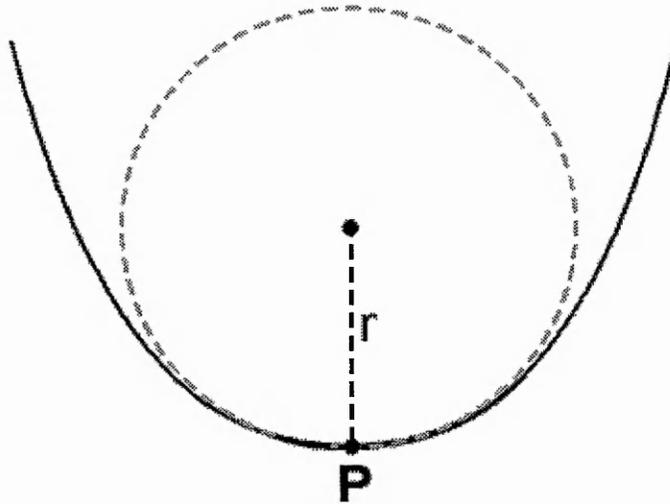
[Jiang & Bunke 94] employ a split-and-merge approach for the fast identification of straight-line segments within individual scan lines, which are then combined in order to extract planar segments. Despite their efficiency these approaches cannot be applied directly to triangulated surfaces because of their lack of an underlying regular grid structure. Therefore a novel method for the fast identification of planar segments in triangulated surface data has been developed and is presented in Chapter 5.

## **2.3 Curvature-based segmentation**

Curvature as an element of differential geometry is often used to gain preliminary information about the surface quality. However, in the majority of the literature curvature is applied to smooth rather than to polyhedral surfaces. As a brief general introduction this section explains curvature for smooth surfaces and shows how to employ it for surface segmentation.

### **2.3.1 Explanation of curvature for smooth surfaces**

Before a formal definition is given, the idea of curvature shall at first be exemplified on 2D curves and then on 3D surfaces. So for a point  $\mathbf{P}$  on a smooth 2D curve the curvature is informally defined as the inverse of the radius of an osculating circle that locally coincides with the curve in a neighbourhood of  $\mathbf{P}$  (see Figure 2.3.1). The meaning of curvature at a point  $\mathbf{P}$  on smooth surface in 3D space can be extended by using perpendicular cross-sections. Such a perpendicular intersection of a plane with the surface contains the surface normal



**Figure 2.3.1:** *The curvature  $\kappa = 1/r$  of a smooth curve at a point  $P$  can be determined from an osculating circle of radius  $r$  that approximates the curve locally in  $P$*

$\mathbf{n}$  and yields a curve in two dimensions whose curvature can be determined as before. Since the intersection plane can be rotated round the surface normal  $\mathbf{n}$  at  $P$ , this procedure provides an infinite number of 2D curves passing through  $P$ . This in turn results in an in general infinite number of corresponding curvature values distributed in a closed interval  $[\kappa_{\min}, \kappa_{\max}]$ . The lower and the upper bound of this interval,  $\kappa_{\min}$  and  $\kappa_{\max}$ , are commonly called **principal curvature values** or less formally **principal curvatures**. These values is given a sign as explained in more detail in Chapter 3. They are of particular relevance for surface analysis, as will be explained shortly. It is also convenient to introduce the **principal radii of curvature**  $R_1$  and  $R_2$  where

$$R_1 = \max\{1/\kappa_{\max}, 1/\kappa_{\min}\} \quad (2.9)$$

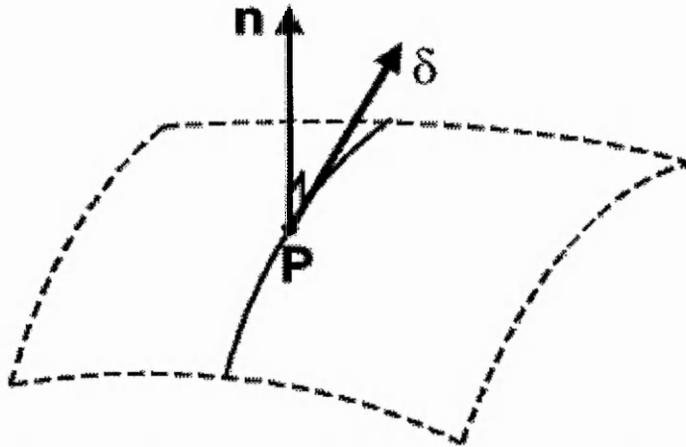
and

$$R_2 = \min\{1/\kappa_{\max}, 1/\kappa_{\min}\}. \quad (2.10)$$

If  $\kappa_{\min}$  and  $\kappa_{\max}$  are both positive and  $\kappa_{\min} \leq \kappa_{\max}$ , then Equations (2.9) and (2.10) imply  $R_1 = 1/\kappa_{\min}$  and  $R_2 = 1/\kappa_{\max}$ . If one of the values  $\kappa_{\min}$  and  $\kappa_{\max}$  is 0, then the corresponding radius is allowed to be infinite. Moreover, each radius obtains the sign from the corresponding curvature value.

Also associated with each curvature value at a point  $P$  on a smooth surface is a normalised directional vector  $\delta$ . It is a vector in the perpendicular cross-section

plane through  $\mathbf{P}$  for which the curvature value is calculated and perpendicular to  $\mathbf{n}$  (see Figure 2.3.2). Hence  $\delta$  lies in the tangent plane of  $\mathbf{P}$ .



**Figure 2.3.2:** Curvature direction  $\delta$  on a smooth surface perpendicular to the corresponding surface normal  $\mathbf{n}$  that is in the tangent plane of a point  $\mathbf{P}$

The directional vectors  $\delta_{\min}$  and  $\delta_{\max}$  that correspond to  $\kappa_{\min}$  and  $\kappa_{\max}$ , respectively, are usually called **principal curvature directions** and are mutually perpendicular. An example of the principal curvature directions at a point  $\mathbf{P}$  on a convex cylinder is presented in Figure 2.3.3. Please note that each curvature direction  $\delta$  is ambiguous since the reversed vector  $-\delta$  has the same properties. However, in practice this ambiguity can be removed by postulating each  $\delta$  to have its largest component positive, and in case of multiple equal components one can be given a preference.

According to [Bronstein & Semendjajew 85] principal curvature values can be determined a little more formally as follows (the German text has been translated by the Author):

*“For a fixed point  $\mathbf{P}_0$  on a surface it is always possible to select a Cartesian coordinate system whose origin is  $\mathbf{P}_0$  and whose  $x,y$ -plane coincides with the tangent plane through  $\mathbf{P}_0$  (see Figure 2.3.4). In this  $x,y,z$ -system the surface possesses (in a neighbourhood of  $\mathbf{P}_0$ ) a representation  $z = z(x, y)$ , where*

$$z(0,0) = \frac{\partial z(0,0)}{\partial x} + \frac{\partial z(0,0)}{\partial y} = 0.$$

*The corresponding accompanying tripod [remark of the Author: better known as “Frenet frame”] in  $\mathbf{P}_0$  consists of the three unit normal vectors  $\mathbf{e}_1, \mathbf{e}_2, \mathbf{N} = \mathbf{e}_1 \times \mathbf{e}_2$ , which point into the directions of the coordinate axes. The Taylor expansion in a neighbourhood of  $\mathbf{P}_0$  is given by*

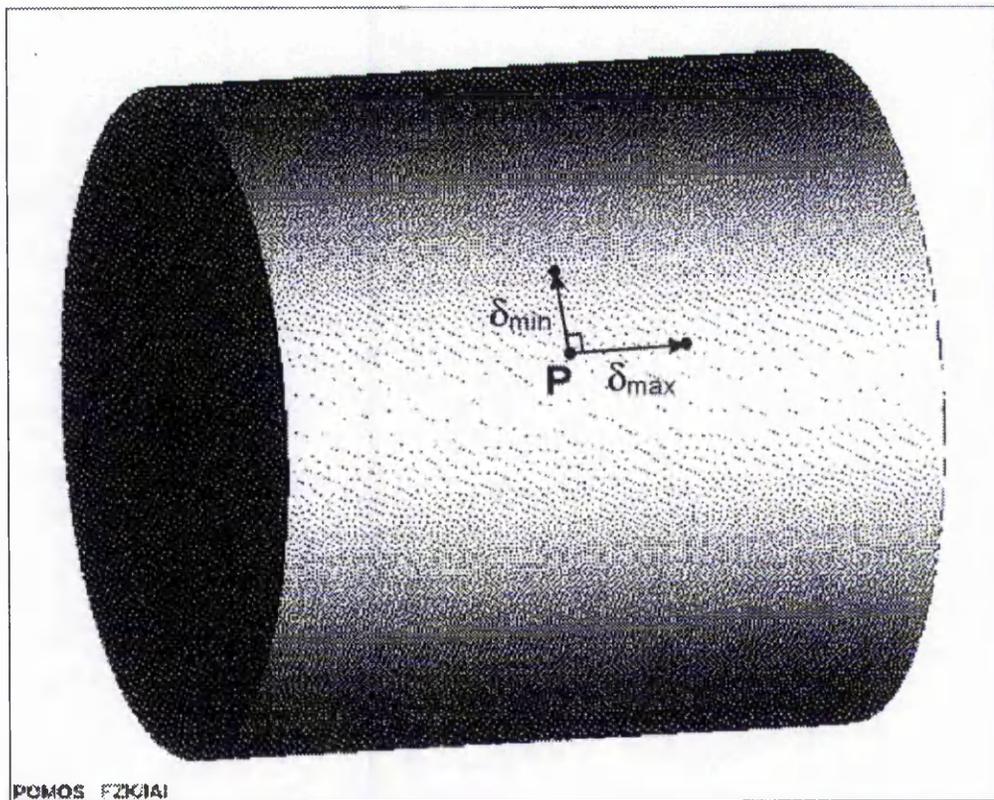
$$z = \frac{1}{2} \frac{\partial^2 z(0,0)}{\partial x^2} + \frac{\partial^2 z(0,0)}{\partial x \partial y} + \frac{\partial^2 z(0,0)}{\partial y^2} + \dots$$

By a rotation of the Cartesian coordinate system around the z-axis one can obtain

$$z = \frac{1}{2} (\kappa_1 x^2 + \kappa_2 y^2) + \dots$$

One defines:  $\kappa_1, \kappa_2$  principal curvatures,  $R_1 = 1/\kappa_1, R_2 = 1/\kappa_2$  radii of principal curvature,  $K = \kappa_1 \kappa_2$  Gaussian curvature,  $H = \frac{1}{2} (\kappa_1 + \kappa_2)$  mean curvature in  $P_0$ ."

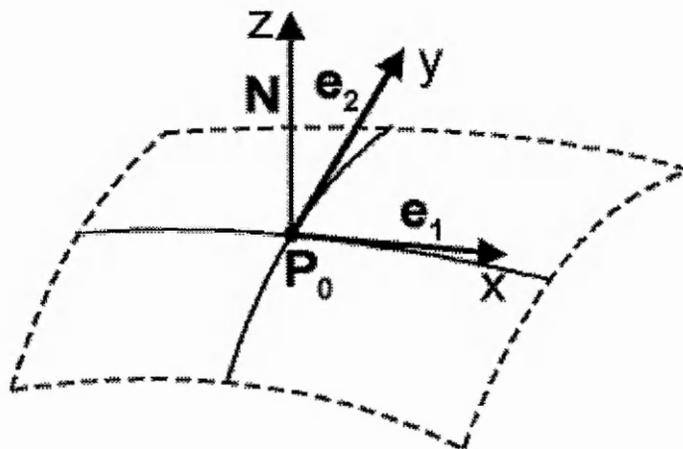
An important property of curvature is **scale-dependence**, i.e. a change of the curvature values in magnitude caused by the multiplication of the points that represent a digital curve or a surface model by a scale-factor. Since curvature depends reciprocally on the radius of an osculating circle (see Figure 2.3.1), the curvature needs to be scaled by the reciprocal scale-factor of the digital model.



**Figure 2.3.3:** Principal curvature directions  $\delta_{max}$  and  $\delta_{min}$  at a point  $P$  on a convex cylinder with corresponding curvature values  $\kappa_{max} = 0$  and  $\kappa_{min} < 0$  (not shown). It should be noted that principal curvature is defined to be negative on convex surfaces (see Chapter 3).

An entirely formal definition of curvature usually involves elements of differential geometry called “fundamental forms”. Since such elements are of little practical use for polyhedral - in particular triangulated - surfaces, further details will not be considered here but may be found, for example, in [Bronstein & Semendjajew 85].

The curvature determination methods presented so far apply to smooth surfaces only. Methods for estimating curvature on triangulated surfaces will be presented in Chapter 3.

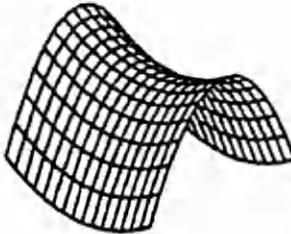
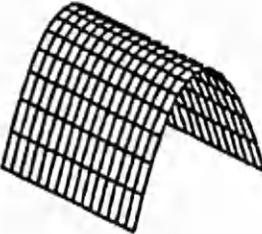
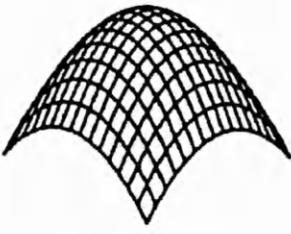
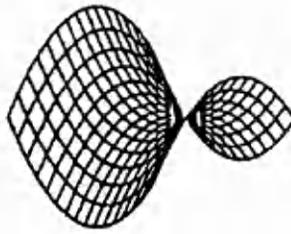
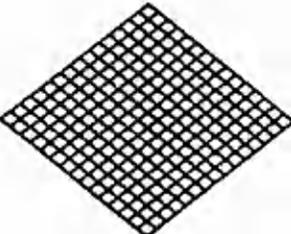
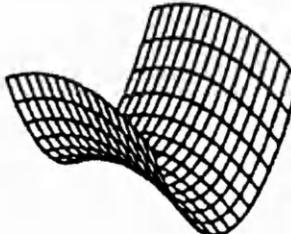
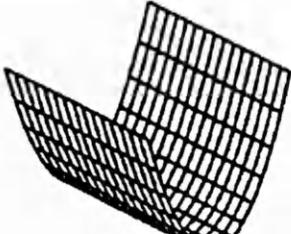


**Figure 2.3.4:** Local  $x,y,z$ -coordinate system in the point  $P_0$  on a smooth surface consisting of the three unit normal vectors  $e_1, e_2, N = e_1 \times e_2$  (taken from [Bronstein & Semendjajew 85])

### 2.3.2 Application of curvature to the segmentation task

Many methods of surface segmentation represented by range images involve the use of curvature to obtain preliminary information about the surface quality (e.g. [Hoffman & Jain 87] and [Trucco & Fisher 95]). For this purpose curvature values have a sign as well as a magnitude in order to distinguish between convex and concave regions such as, for example, the outside of a hemisphere and the inside of a hollow hemisphere. Because for a range image these values depend on the viewpoint of a range image, it is convenient to apply the previously introduced mean (denoted by  $\mathbf{H}$ ) and Gaussian curvature (denoted by  $\mathbf{K}$ ) values. As [Besl & Jain 88] explain these values are viewpoint independent, i.e. “invariant to rotations, translations, and changes in parameterization”. So by means of the sign of the mean and Gaussian curvature values eight fundamental surface types

can be distinguished as shown in Table 2.3.2.1 (see Figures 1 and 3 in the same reference):

	$K < 0$	$K = 0$	$K > 0$
$H < 0$	Saddle ridge 	Ridge 	Peak 
$H = 0$	Minimal surface 	Flat 	Not possible
$H > 0$	Saddle valley 	Valley 	Pit 

**Table 2.3.2.1:** Fundamental surface types associated with the sign of Gaussian ( $K$ ) and mean ( $H$ ) curvature (adapted from [Fischer 99])

Curvature can only be calculated for a smooth underlying surface. A smooth surface patch can be obtained by interpolation of a set of connected data points by a B-spline or NURBS surface. Sometimes digital surface data is then segmented according to the sign of  $H$  and  $K$  values (e.g. in [Trucco & Fisher 95]), but more often these values guide the determination of initial seed selection for region growing, in particular [Besl & Jain 88] and papers referring to it.

Once curvature has been determined for all points on a surface the next step is to find extremal points, i.e. points where the surface bends the most. Connecting these points may yield ridge lines, which in turn may serve as the boundary of surface segments such as in [Chen & Liu 97] or [Lukács et al. 98]; this is also the idea of the “feature lines” in [Hoschek et al. 98]. Moreover, [Tanaka et al. 98] employ principal curvatures and directions for face recognition, where they use

“spherical correlation”. Each face in both input image and a model database is represented as an “extended Gaussian image” (EGI) that is constructed by mapping principal curvatures and directions onto two unit spheres, each of which represents ridge and valley lines, respectively. An individual face is then recognised by comparing similarities among others applying Fisher’s spherical correlation to the EGI’s of faces.

Problems with the determination of curvature by the local interpolation of a smooth surface result mainly from inaccurate data (particularly in [Bolle & Sabbah 87]), i.e. spurious data points impair the curvature estimates. However, this problem is not too grave as the surface may be smoothed beforehand. The latter reference shows that synthetically generated planes, spheres, cylinders, and cones can already be classified by means of curvature.

#### **2.4 Region growing as an efficient bottom-up approach for range images and triangulated surfaces**

The concept of region growing as an approach for surface segmentation has been introduced in 2.2.3. Now a basic region growing algorithm suitable for range images based on a functional surface representation shall be presented following [Taubin 91]. For this an increasing sequence  $F_1 \subseteq \dots \subseteq F_{\max\_order}$  of families of functions is supposed to be given where  $F_1$ , for example, is the family of first-degree polynomials. Furthermore, a region is defined as a data structure  $R = (S, f, order)$  (where  $S$  is a connected subset of data points), and  $f$  is an element of  $F_{order}$  that approximates every point of  $S$  “well”. Now, after initialising  $order$  with 1, the algorithm is:

1. Find a seed region  $R = (S, f, 1)$  where  $S$  is a subset of data points, and use a member  $f \in F_1$ , whose set of zeros approximates every point in  $S$  well.
2. For a given a current region  $R = (S, f, order)$  repeat, until no further growth in  $S$  occurs:
  - a. Compute maximal connected region  $S'$  of points well approximated by  $f$  and intersecting the initial seed set.
  - b. If  $size(S') = size(S)$ , then exit loop.
  - c. Fit new member  $f' \in F_1$  to  $S'$

- d. If  $f'$  satisfies a “goodness of fit” test, then
    - Replace R by  $(S', f', order)$ , go to **2a.**
    - Else
    - Exit loop.
- 3.** If  $order = max\_order$ , then
- Terminate algorithm and return current region  $R = (S, f, order)$ .
  - Else
  - a. Fit a member  $f'$  of  $F_{order+1}$  to S
  - b. If  $f'$  satisfies the “goodness of fit” test in **2d.**, then
    - Replace  $f$  by  $f'$
    - Increase  $order$  to  $order+1$
    - Replace R by  $(S', f', order+1)$
    - Go to **2a.**
  - Else
  - Terminate algorithm and return current region  $R = (S, f, order)$ .

Several steps of the algorithm (such as the initial estimation of characteristic parameters, adaptive surface, and parameter fitting) may require further refinement and are discussed in the subsequent sections.

### 2.4.1 Estimates of characteristic parameters

[Ng et al. 95, Fig. 1 and 2] describe how to derive parameter estimates for cone and cylinder from a pair of surface patches though no details are given showing how this has effectively been accomplished. More details about obtaining characteristic parameters for geometric primitives may be found in [Lukács et al. 98]. Firstly, surface normal vectors are determined by any local surface fitting method. Next, for cylinder, cone, and torus an axis of rotation is estimated based on these surface normals by solving a linear system of equations which arises from the following problem: “*given  $m$  straight lines in three dimensions, compute the straight line intersecting all of them (if such a line exists).*” This is based on the fact that on a surface of revolution the normals intersect the rotational axis. Parameters for cylinder and cone are then obtained using the polar coordinate representation of a normal vector  $\mathbf{n}$ :

$$\mathbf{n} = (\cos \varphi \sin \vartheta, \sin \varphi \sin \vartheta, \cos \vartheta). \quad (2.11)$$

Here  $\vartheta$  denotes the angle between  $\mathbf{n}$  and the  $z$  axis and  $\varphi$  the angle of the projection of  $\mathbf{n}$  onto the plane  $z = 0$  with the  $x$  axis (note that, although such a representation seems attractive since it decreases the number of parameters, it is subject to ambiguities, e.g. when  $\mathbf{n}$  is parallel to the  $z$  axis). By further geometric calculations the authors determine relatively easily the remaining characteristic parameters of cylinder and cone, whereas for the torus they state, “*more robustly, one can opt to estimate principal curvatures of the surface at the base point*” [remark: base point = point at which an estimate of the surface normal vector exists].

### 2.4.2 Surface fitting

As is shown in [Roth & Levine 93] the task of fitting a surface to a set of given data points  $\Omega$  is equivalent to find the optimum value of a “cost function”, for example a function that represents the quality of fit. This quality of fit can be quantified by the overall distance function  $d(\Omega, \mathbf{X})$  defined in Equations (2.7) and (2.8), respectively. The minimisation of the overall error  $d(\Omega, \mathbf{X})$  with regard to the parameter vector  $\mathbf{X}$  forms an optimisation problem, which is in general non-linear owing to the functions that are involved. Depending on these, the problem may have multiple minima. Consequently an appropriate optimisation algorithm requires, besides a good performance, the ability to escape from such minima in order to detect a global minimum.

Furthermore, it would be desirable if such an algorithm could perform constrained optimisation. For example, if a vector  $\mathbf{X}$  of characteristic parameters of a geometric primitive, e.g. the axis vector  $\mathbf{a} = (a_1, a_2, a_3)$  of a cylinder, requires normalisation, this can be expressed using a constraint function  $c(\mathbf{X})$  of the form

$$c(\mathbf{X}) = (a_1^2 + a_2^2 + a_3^2)^{1/2} - 1 = 0 \quad (2.12)$$

where  $\mathbf{X} = (\mathbf{C}, \mathbf{a}, r)$  (see Section 4.3.3). The two optimisation problems using expressions (2.7) and (2.8) are referred to as - unconstrained or constrained - least-squares problem (LSP) and minimax problem (MP), respectively.

In order to solve a general minimisation problem analytically, the first derivative of the objective function needs to be determined. After this expression is set to 0, the resulting equation needs to be solved with respect to the unknown vector  $\mathbf{X}$  of characteristic parameters. Essentially the same approach, though with a higher number of unknowns, can be applied to the above LSP, whereas for the MP this is not possible, since neither the maximum function nor the absolute value function is differentiable.

As is reflected by the literature, the majority of researchers deal with the LSP but some different approaches are also known. [Roth & Levine 93] introduce the “minimal subset principle” based on the observation that a geometric primitive through a “minimal subset of points” is often a good description of all the points fitted by the geometric primitive. Here the “minimal subset of points” refers to “*the smallest subset necessary to produce a unique instance of a geometric primitive*”. Based on the minimal subset principle several techniques have been developed in order to identify an appropriate point subset. [Roth & Levine 90] employ a Genetic Algorithm (GA), which initially takes a number of random seed points and determines from these in a stochastic manner (emulating the evolutionary principles of mutation, crossover and “survival of the fittest” as for natural genes) an adequate minimal subset by successive evaluations of the cost function. For each of these evaluations it is required to deduce the primitive parameters from the minimal subset.

Similarly, [Ke et al. 97] use “tabu search” in order to identify a suitable minimal subset. Like a GA, tabu search has the ability to escape from local minima. Starting from an “initial minimal subset of points”, a new point set is obtained by successive replacement of some points in the expectation to obtain a minimal subset of better fit. In contrast to the GA approach, tabu search restricts the search space by eliminating solutions that were found previously, and these are stored in a “tabu list”. Even a temporary “bad solution” (i.e. an inadequate minimal subset) in an intermediate phase may be used as a basis for determining a replacement of points such that the corresponding minimal subset yields a better set of representatives. For this reason tabu search can also escape from local minima.

So far GA and tabu search have been employed only for the extraction of 2D geometric primitives such as circle and ellipse, whereas for 3D primitives these algorithms are still subject of current research. Their advantage lies in their **robustness**, which means both algorithms can process highly noisy data. However, these algorithms perform the extraction of higher order primitives rather slowly with respect to the amount of data required for reverse engineering applications because of the currently available computational power. The determination of characteristic parameters of a geometric primitive for both optimisation methods requires at present computer algebra (“Gröbner basis techniques”), so that it would be difficult or even impossible to perform online-calculations without using a dedicated software system. Moreover, data obtained from most commercial 3D scanning systems for reverse engineering shows in general a good accuracy. Therefore dedicated numerical optimisation methods are deemed to be more favourable for primitive extraction since they attempt to solve LSP and MP directly rather than determining minimal subsets representing the primitives.

As stated above numerical methods for solving MP must be able to determine the minimum of the objective function without information about derivatives. They can be categorised into **directional** and **non-directional** methods. In general, both start the minimisation from an initial guess  $\mathbf{x}_0$ , and then attempt to improve it iteratively by determining a sequence of parameter vectors  $\mathbf{x}_i$  of decreasing values with respect to the objective function  $f$  which means  $f(\mathbf{x}_{i+1}) < f(\mathbf{x}_i)$ . Basically they differ in the way  $\mathbf{x}_{i+1}$  is obtained from  $\mathbf{x}_i$ . Non-directional methods such as the GA or tabu search try to find a  $\mathbf{x}_{i+1}$  without particular exploitation of  $f$  itself. They simply evaluate a function  $f$  for two (or more) subsequent parameter vectors and compare the results in order to ensure the decrease of the function, while  $\mathbf{x}_{i+1}$  is essentially deduced from  $\mathbf{x}_i$  in a stochastic manner. For this reason non-directional methods are fairly robust to “non-smoothness” of a function  $f$  but very time-consuming in their implementations.

On the contrary, directional methods take the special structure of a function  $f$  into account. For the purpose of minimisation often a **steepest descent** direction  $\mathbf{d}$  of  $f$  is searched in the hope that the graph of  $f$  will decrease in this direction for a sufficiently long distance in parameter space, i.e. for a parameter vector  $\mathbf{x}$  it is attempted to minimise  $f(\mathbf{x} + \lambda \mathbf{d})$  by increasing the value for  $\lambda > 0$ . Such a

direction of steepest descent may be obtained either by differentiation, if  $f$  is differentiable, or as a composite direction obtained from predetermined search directions. Optimisation algorithms not involving derivatives, sometimes denoted as “direct search algorithms” ([Powell 98]), usually require more evaluations of the objective function in order to determine the steepest descent, which are likely to deteriorate the algorithmic performance. On the other hand the computation of derivatives of the objective function is also time-consuming, so that in general no statement can be made which algorithm to prefer. Here other aspects such as numerical stability are disregarded. In clear contrast to non-directional methods, directional approaches can terminate prematurely in a local minimum (which can be imagined as a valley in the graph of the function) rather than a global minimum (corresponding to a deepest valley; there may be more than one; see also Figure 4.4.1). Hence the initial determination of a characteristic parameter vector close to the final solution is a crucial task for optimisation problems in general and for surface fitting in particular.

In general literature on optimisation problems exists widely (e.g. [Gill et al. 81], [Stoer 83], [Fletcher 87]) though least-squares problems are more often considered than minimax problems. Some of the few dealing with this type of problem do not reveal all details necessary for a successful implementation (e.g. [Hald & Madsen 81] and [Polak et al. 92]). However, even the detailed algorithm given in [Charalambous & Conn 78] requires for certain problem types a “good initial guess” as a starting vector, i.e. a vector in the domain of convergence. It is still an open problem how to determine such a “good initial guess” for an appropriate starting vector in general, because not even criteria for the goodness of the guess have been established.

### **2.4.3 Segment extraction**

Fitting a certain type of a geometric primitive to a set of adjacent points is only one problem in surface segmentation; determining the most appropriate type of such a primitive for fitting is another one. A type of geometric primitive is considered as more appropriate than another if it requires a smaller vector of characteristic parameters and provides a larger area of fit. In practice these two qualities need to be weighted and combined together with a measure of the

goodness of fit into a single criterion that allows a comparison between fits of primitives of different types and characteristic parameters.

To achieve all segmentation aims (namely the proper selection of a type of geometric primitive for approximating a surface, a good fit, and a large number of points being fitted) [Leonardis et al. 95] employ a “recover-and-select paradigm”. In the first phase data aggregation is performed via model recovery using variable order bivariate polynomials obtained from iterative regression. For the model recovery seed regions are regularly placed independently in the image. After a step of limited region growing for each seed, the best models serve as candidates for the final description of the data. Then, after a step of unrestricted region growing, the best model is determined by solving a quadratic Boolean problem that emerges from three quantities: number of data points explained by the selected model, number of parameters required for its description, and a measure of deviation between model and data (i.e. a measure of the goodness of fit).

Despite a good tradeoff between algorithmic performance and quality of extracted segments the recover-and-select paradigm presents some difficulties when applied to triangulated surfaces. For example, the seed regions are obtained from a grid of windows overlaid on the image. Such non-overlapping seeds can be defined straightforwardly on a range image as the data points are parametrised. On a triangulated surface, however, such a non-overlapping seed definition requires an initial determination of connected vertices or triangles as seeds. Another relevant aspect for the definition of seed regions concerns the grid size which can be varied easily on a range image, but not on a triangulated surface. However, it is not clear how seeds in range images are determined for regions of low data point density, a problem that has an equivalence in regions of relatively large-sized triangles on triangulated surfaces.

For the above reasons the recover-and-select paradigm is difficult to apply to triangulated surfaces, whereas the method of [Besl & Jain 88] promises an easier transferability, which will be discussed next. In principle, the method consists of two stages, namely “Sign-of-Curvature Detection” and “Surface Primitive Extraction”. During “Sign-of-Curvature Detection” each pixel in an image is given a label referring to one of the fundamental surface types shown in Table 2.3.2.1. Of course, this stage involves a prior determination of curvature values.

For the “Surface Primitive Extraction” iterative region growing is performed based on variable-order surface fitting. Initially the measurement accuracy of the data is estimated by a local equally-weighted least-squares planar fit in the 8-connected neighbourhood of a pixel (including the pixel). From this accuracy threshold values for subsequent tasks are deduced. Seed regions are found by repeated application of an “erosion operator” on connected regions of a fundamental surface type until the number of pixels in each region falls below a certain threshold. These seed regions are then allowed to grow, where initially the order of the polynomial function that models the region is low, but may increase during the growing process if the fit is not satisfying. This principle is what Besl and Jain refer to as “variable order surface fitting” (compare to [Taubin 91] at the beginning of this section). Region growing terminates if the growth process yields a constant region size or the highest order function cannot approximate the image data. At the end of each growth step connected regions are validated, and either accepted or rejected depending on the “goodness of fit” of its underlying polynomial function. Its corresponding pixels are marked so that they are not considered for future seed regions.

One advantage of this approach is that it can be performed in parallel: every seed region may be allowed to grow simultaneously, where in each step a list of “compatible” pixels of a region is determined (see [Besl & Jain 88] and [Leonardis et al. 95]). Then without further postprocessing operations non-adjacent compatible regions can be labelled as such during the step of region validation (“surface acceptance stage”).

The method of [Besl & Jain 88] can only be applied to triangulated surfaces with some restrictions since it uses bivariate polynomials for the representation of digital surfaces that are parametrised. This is mainly because such a representation simplifies curvature computations, but on the other hand it cannot model a torus. Further limitations for the application to triangulated surfaces result from the use of “window operators” (such as a  $3 \times 3$  least-squares planar surface fit operator). [Leonardis et al. 95] pursue a similar approach insofar as they also use bivariate polynomials. Their work differs to [Besl & Jain 88], however, insofar as they employ region growing to

*“independently build all possible models using all statistically consistent seeds, found in a grid of windows overlaid on the image, and then use the recovered*

*models as hypotheses that could compose the final description. To determine the statistical consistency of a seed, we fit a model to the data points in the seed window*".

The principle of this "recover and select paradigm" has also been used in [Lukács et al. 98] in which the surface fitting is simplified by a linear approximation of the point-to-surface distance. So all of these methods require the determination of a complete (and sometimes disjoint) window-like partition of a triangulated surface, which appears to be a fairly complicated task, if it can be achieved at all.

Very closely related is also the work of [Fisher et al. 97], where complete surface patches are extracted from a triangulated surface obtained from multiple registered range images. Further extraction stages consist of "local curvedness estimation and shape classification", "surface patch growing", and "patch edge adjustment". Although it is stated in this paper that "*it is easy to estimate local surface curvatures at each point through a local surface fitting algorithm*", details are not presented. Moreover, because the surface representation used there is based on quadrics, toroidal surfaces are not extracted.

## **2.5 Open problems**

In engineering and design applications surfaces of simple shape play an important role. Often it is desired to approximate an existing triangulated surface by parts of geometric primitives. Commonly the set of employed geometric primitives encompasses planes, spheres, cylinders, cones, and tori. Explicit functions cannot represent a complete sphere because a spherical surface cannot be mapped globally onto a plane without loss of the topology. This drawback can be overcome by using an implicit representation for the surface of a geometric primitive.

Previous definitions of the segmentation problem have been formulated for range images with synthetic free data without taking noise in the data into account. Such definitions are suitable under the assumption that the data is "ideal", or otherwise they rely on the trivial "one pixel per segment" solution. Moreover, in the previous problem definition each segment consists of individual pixels rather than of triangles but unlike for pixels the intersection of adjacent triangles that belong to different segments is not empty. However, independent from its

representation measured non-synthetic data is affected by noise, and thus an appropriate problem definition needs to take this also into account.

For the segmentation of a range image top-down approaches have emerged to be useful since the data is parametrised. As triangulated surfaces do not allow for a global parameterisation these approaches present enormous difficulties when applied to such surfaces. Most of the previous bottom-up approaches also have assumed the availability of a surface parameterisation but [Fisher et al. 97] show that a triangulated surface can be segmented without this assumption. However, many of them assume an *a priori* grouping of the data points in order to identify the individual groups, which is a simpler task than to determine an appropriate grouping.

Clustering techniques may well be able to segment a surface but the segments obtained from this technique are most likely to be unconnected. Moreover, because of the noise in the data, a segmentation based on such techniques is expected to be fairly coarse. Segmentation techniques that are based on edge-detection suffer from the drawback to be unable to detect smooth joins between surface segments such as, for example, a smooth join between sphere and cylinder. A further essential drawback of edge-detection and similar methods is that they provide only oriented boundary curves enclosing a detected segment but neither information about shapes (i.e. parts of geometric primitives) nor characteristic parameters. Stochastic segmentation techniques offer the benefit of yielding approximate parameter values for each extracted geometric primitive regardless of its type. However, these techniques (such as the Hough transform) are computationally inefficient, particularly for processing triangulated surfaces with a large number of triangles. This is because the Hough transform either does not yield very accurate results, or the corresponding algorithm consumes ever more computational memory when refining the discretisation. Combined approaches also rely mostly on a surface parameterisation and have mainly been developed for the identification of planar regions in range image data.

As a useful surface-based property mean and Gaussian curvature can be calculated from a smooth polynomial that locally approximates the parametrised surface at a point of a range image. So a curvature sign map can be applied to obtain a segmentation in terms of eight fundamental surface types. Furthermore,

curvature can guide the selection of suitable seeds for a bottom-up segmentation approach known as region growing.

Although region growing presents some difficulties when applied to triangulated surfaces, these are of general nature and not directly related to this data representation. The individual problems to solve concern the estimates of characteristic parameters, the fitting of an appropriate candidate surface, and the identification of the most appropriate surface segment to extract. Examining the previous work for all these tasks offers opportunities for improvements. In particular only few methods for the determination of characteristic parameters are given in full detail in the current literature. One of these methods presented in [Lukács et al. 98] requires the solution of a linear system of equations which may be numerically instable. Consider, for example, the radius of a large cylinder where only a few normal vectors are known on adjacent triangles. Then the normals of the cylinder are almost parallel to each other and thus the system of equations is likely to be ill-conditioned.

## **2.6 The approach used in this project: curvature estimation and region growing**

### **2.6.1 Representation of geometric primitives for surface segmentation**

For what follows it is convenient to define the set

$$\mathbf{G} = \{\text{pl, sph, cyl, con, tor}\} \quad (2.13)$$

of geometric primitives representing plane, sphere, cylinder, cone, and torus, respectively, which are subject of the remainder of this thesis. It shall be noted that [Lukács et al. 98] employ basically the same set of geometric primitives - except planes - for surface classification rather than using bivariate polynomials or (super-)quadrics.

The surface of each geometric primitive  $g \in \mathbf{G}$  is represented by the implicit equation

$$f_g(\mathbf{P}; \mathbf{X}_g) = 0, \quad (2.14)$$

where  $f_g$  represents the Euclidean distance from a point  $\mathbf{P}$  to  $g$  (see appendix A for the derivation of  $f_g$  for all  $g \in \mathbf{G}$ ) and  $\mathbf{X}_g$  denotes an appropriate parameter vector that characterises the primitive. However, for noisy data points the requirement in Equation (2.14) is too rigorous. Thus each point  $\mathbf{P}$  belonging to  $g$  is assumed merely to meet the condition

$$f_g(\mathbf{P}; \mathbf{X}_g) < \tau, \quad (2.15)$$

where  $\tau > 0$  is an appropriate pre-defined tolerance that may depend, for example, on the measurement accuracy of a given set of data points.

Using the above notation Section 4.1 reformulates the specification of the segmentation problem so that it is applicable to triangulated surfaces rather than range images. Furthermore, the reformulated problem specification takes measurement errors of the data points into account and is therefore based on the use of a tolerance value  $\tau$ .

The problem remains how to choose  $\tau$  properly because for  $\tau = 0$  a segmenter is likely to split the entire surface into individual data elements, whereas for  $\tau$  sufficiently large (i.e. larger than the size of the object) the segmentation result would yield the originally input surface. A possible way how to determine a proper value for  $\tau$  automatically from the data (as an “adaptive tolerance”) is presented in Section 5.2.2.

## 2.6.2 Problem formulation

Most of the previous segmenters ([Besl & Jain 88], [Biswas et al. 95], [Chen & Liu 97], [Chen & Liu 99], [Han et al. 87], [Hoffman & Jain 88], [Hoover et al. 96], [Jiang & Bunke 94], [Ke et al. 97], [Leonardis et al. 95], [Maître et al. 90], [Mirza 95], [Ng et al. 95], [Powell 98], [Roth & Levine 93], [Taylor et al. 89], [Trucco & Fisher 95], [Werghi et al. 99], [Yokoya et Levine 91], [Yu et al. 94], [Zhao & Zhang 97]) are based on the approaches and techniques introduced in Section 2.2. Consequently, this project tries to combine and transfer their principles (wherever possible and appropriate) in order to achieve a complete segmentation of a triangulated surface and extraction of parts of geometric primitives. The most promising ideas seem to be curvature and region growing.

After introducing the usefulness of curvature for smooth surfaces in Section 2.3 some existing curvature approximation methods more dedicated to triangulated surfaces are presented in Chapter 3. Moreover, the drawbacks of these methods are discussed. In order to avoid them a novel method of curvature estimation for such surfaces is established that requires triangle normals and adjacency information only. Afterwards various methods will be compared to each other, and some applications to “real” data are shown.

It is recalled from Section 2.4 that problems associated with region growing arise mainly from the estimation of characteristic parameters, surface fitting, and segment extraction. Because of limitations in both software and hardware the focus of this project is set on successive region growing rather than employing a concurrent approach.

In Section 4.2 a modified region growing algorithm used for this project is presented. In particular surface fitting and feature segment extraction is explained in more detail, where the fitting problem involves a readjustment of the characteristic parameters (numerical optimisation) for each type of geometric primitive during region growing. A two stage optimisation method combining a genetic algorithm and a direct search method show better fitting results than each of the two optimisation methods alone. Furthermore, the section offers a strategy for the identification of appropriate segments that are extracted from a triangulated surface.

The few previous methods ([Lukács et al. 98], [Ng et al. 95]) for the determination of characteristic parameters of geometric primitives are either not listed in full detail required for a successful implementation, or they involve the solving of a system of linear equations, which is likely to be numerically unstable in certain practical cases. Therefore, Section 4.3 presents a simple geometric and, to the best of the Author’s knowledge, novel method for the estimation of such characteristic parameters that is based on the curvature estimates presented in Chapter 3.

The segmenter is then tested in Section 4.6 on triangulated surfaces that contain components of various geometric primitives, where the data points are affected by measurement errors in an order of magnitude that is usual for most of the current

conventional 3D scanning systems. Results are presented to show the quality of the initially estimated characteristic parameters as well as the extracted segments.

Planar regions play an important role in CAD/CAM applications (e.g. in [Ashbrook et al. 97]). Therefore, Chapter 5 is dedicated to a novel technique for the automated extraction of planar segments from triangulated surfaces based on fairly simple geometric considerations. In clear contrast to many previous techniques, the presented one requires only adjacency information for triangles, which is exemplified on triangulated data with a “usual” amount of noise. Additionally, an extension of this technique offers an estimation of the accuracy of the scanned data.

Chapter 6 discusses not only the achievements, but also the limits of the novel methods and techniques proposed in this thesis. Furthermore, future work is outlined that potentially leads to an improved segmentation of triangulated surfaces. Conclusions are given in Chapter 7.

### **3 Curvature estimation for triangulated surfaces**

The estimation of curvature on a triangulated surface is able to provide a segmenter with information about surface regions that resemble parts of particular geometric primitives. For instance, a planar segment may be found where the surface shows almost zero curvature, and a spherical segment where the surface shows nearly constant non-zero curvature. Therefore an initial estimation of curvature supports the selection of a set of seed points according to the bottom-up segmentation strategies discussed in Section 2.2.3. Furthermore, principal curvature values and directions offer the possibility of calculating estimates for the characteristic parameters of the geometric primitives in  $\mathbf{G}$  (which is subject of Section 4.3).

The novel “NEN method” for curvature estimation on triangulated surfaces established by the Author is based on a method employing the “difference of normals” used by [Ittner & Jain 85] (so it is referred to by [Flynn & Jain 89]). The novel NEN method compares favourably to existing discrete curvature estimation techniques. Moreover, it is likely to require less computational time than techniques that rely on the fitting of a smooth surface through the data points prior to curvature estimation on this surface.

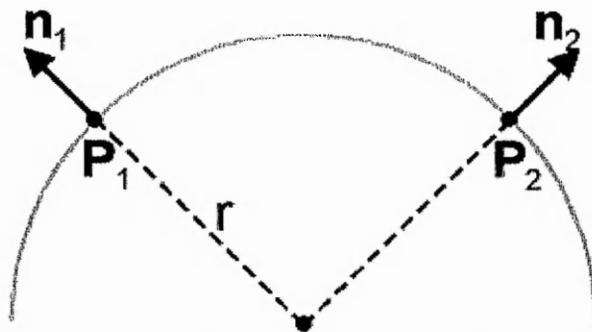
Section 3.1 gives an overview about the few existing methods and their potential drawbacks. Section 3.2 presents the novel NEN method for estimating curvature on triangulated surfaces. For this interpolated and compensated normals for triangulated surfaces are introduced in Section 3.2.1. Section 3.2.2 explains how to use these normals for the estimation of principal curvature values and directions. A set of suitable test data is established in Section 3.3 in order to allow a comparison between the methods. Results of this comparison are given in Section 3.4. Some applications to non-synthetic data are illustrated in Section 3.5. A summary in Section 3.6 concludes this chapter.

#### **3.1 Previous methods of curvature estimation**

Though not especially designed for triangulated data, at first a method based on the “difference of normals” suitable for a set of data points is presented. A

second method estimates “discrete curvature”, and a third one involves curvature approximations by local fits of smooth paraboloid surfaces.

[Flynn & Jain 89] distinguish in their comparison of methods for “digital curvature estimation” between analytic and numerical estimates, where analytic methods encompass orthogonal polynomials, linear regression and spline-based estimates. On the other hand their numerical methods comprise “difference of normals” (short: DN) estimate and directional curvature from derivatives. These five methods were tested on synthetic and real range image data. Their synthetic data is similar to what is presented in Section 3.2 with respect to the type of geometric primitives and the corresponding characteristic parameters. Since [Flynn & Jain 89] conclude that in their tests “*overall the DN (difference-of-normals) method seems to have performed best*” only this method out of several others is considered in the following. Moreover, this method can be integrated to the NEN method introduced in Section 3.2.



**Figure 3.1.1:** Calculation of curvature  $\kappa=1/r$  using two points  $P_1$  and  $P_2$  on a scanned surface with corresponding normals  $n_1$  and  $n_2$  (used by [Ittner & Jain 85])

So suppose there are two points  $P_1$  and  $P_2$  on a scanned surface with corresponding normal vectors  $n_1$  and  $n_2$  such that the lines in the directions of the normals pass through  $P_1$  and  $P_2$ , respectively (see Figure 3.1.1). [Flynn & Jain 89] state the following formula for determining the curvature  $\kappa$  between  $P_1$  and  $P_2$ :

$$\kappa = \frac{\|n_1 - n_2\|}{\|P_1 - P_2\|} \quad (3.1)$$

Henceforth this formula shall be referred to as the **DN (curvature) formula** (its origin is likely to be found in [Ittner & Jain 85]). Geometrically the DN formula

can be interpreted as the angular deviation of the pair of normals  $\mathbf{n}_1$  and  $\mathbf{n}_2$  located at  $\mathbf{P}_1$  and  $\mathbf{P}_2$ , respectively.

In theory the DN formula could be applied to the vertices of a triangulated surface but the problem is that surface normals at these vertices may not be defined. Nevertheless the idea of this method is fitted into the NEN method that provides suitable “interpolated normals” at vertices as presented in Section 3.2. Section 3.4 shows that the application of the DN formula also allows a qualitative comparison to the NEN method. In fact, the data used for a comparison is similar insofar as each synthetic range image used in [Flynn & Jain 89] can be easily converted to a triangulated surface by the POMOS software tool.

Another method (found in [Krsek et al. 98]) for polyhedral surfaces employs “discrete (Gaussian and mean) curvature” (short DC):

*“The discrete Gaussian curvature of a vertex in a triangulation is the angular deficit of the vertex with respect to the sum of angles  $\phi_{ik}$  of the adjacent triangles:*

$$\omega_i = 2\pi - \sum_k \phi_{ik}.$$

*Under certain assumptions, the density of  $\sum_i \omega_i$  with respect to the area of the respective portion of the polyhedron gives a good approximation of the average Gaussian curvature  $K$  over this part of the measured surface. ... Similarly the discrete mean curvature  $\psi(e)$  of an edge  $e$  in a triangulation is half the angle between the plane normals of the faces adjacent to  $e$ . The mean curvature measure of a portion  $U$  of the triangulated surface is:*

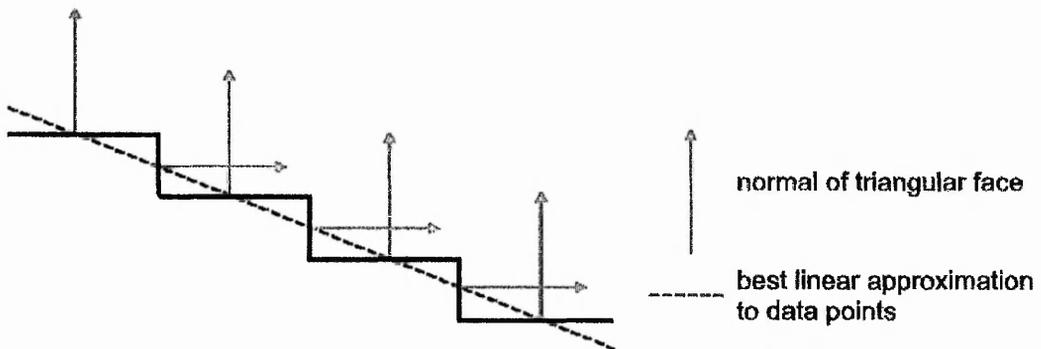
$$\psi(U) = \sum_e \psi(e) \text{ length}(e \cap U).$$

*Similarly  $\psi(U)/\text{area}(U)$  provides a good estimate of the average mean curvature  $H$  over the part of the measured surface which corresponds to  $U$ .”*

After dividing the terms  $\sum_i \omega_i$  and  $\sum_e \psi(e) \text{ length}(e \cap U)$  by the  $\text{area}(U)$  for a portion  $U$  of a triangulated surface they become scale-dependent. However, imagine a perpendicular intersection of, for example, a triangulated plane would yield a zigzag line, as illustrated in Figure 3.1.2, where the edge  $e$  is assumed to be perpendicular to the image. Then such a plane is likely to spoil estimates for the discrete mean curvature unless the term  $\psi(e)$  is provided with a sign to allow compensation for noise in the data, as proposed by [Brehm & Kühnel 81].

Therefore for the following comparison the sign of  $\psi(e)$  has been defined as

positive, if the edge  $e$  is shared by a concave pair of adjacent triangles, and negative otherwise. Without this assumption the estimated curvature values emerged to be worse.



**Figure 3.1.2:** *Perpendicular cross-section of a triangulated surface of a staircase that has face normals which are either parallel or orthogonal to each other*

Another method suitable for estimating curvature for polyhedral surfaces involves “paraboloid fitting” to the data points, i.e. the surface is approximated locally by an analytic surface of second order. This method (described in [Hamann 93]) encompasses several steps as summarised in [Krsek et al. 98]:

*“Estimate the normal by locally approximating the data by a plane.  
 Transform the neighbourhood points into a coordinate system with origin at the given point and  $+z$  along the normal.  
 Fit a second order surface in this coordinate system  $z = f(x, y)$ .  
 Compute curvatures from the parameters of this analytic surface.  
 To be able to do this, the data must form a single valued function in a small neighbourhood of each point.”*

The “paraboloid fitting” requires a second order surface fit at each data point. This necessitates solving a local least-squares problem at each vertex which is as relatively time-consuming. Moreover, in the context of segmenting a triangulated surface into parts of geometric primitives doubts arise concerning its adequacy since [Krsek et al. 98] conclude:

*“The approximation model is a paraboloid. There is thus a systematic error when a sphere or cylinder is approximated.”*

## 3.2 Derivation of the novel NEN method

In a CAD context a triangulated surface may be derived from a scanned image of a possibly smooth physical object, henceforth referred to as **virtual original**. For

such a surface the vertices lie approximately on the virtual original, the difference being due to measurement errors. However, triangle edges and interior triangle points are merely obtained by joining up the vertices. Thus - unlike for smooth surfaces - the calculation of curvature for a triangulated surface itself yields no useful information, because the curvature is zero at points within a triangle, and infinite at points on edges shared by two non-coplanar triangles. So it is more useful to introduce curvature *estimates* on triangulated surfaces. The idea of the estimate aims to represent the curvature of the virtual original, assuming it is smooth. This assumption makes sense because many objects are composed of parts of geometric primitives that have a smooth surface. Therefore the estimates in this thesis are based on simple geometric calculations using the triangles of a surface.

An intuitive way of estimating curvature on triangulated surfaces is the following method, which from now on will be referred to as **triangle-pair method**: for each pair of adjacent triangles compute the centre  $S$  of a sphere passing through their four vertices. Now radius of the sphere is  $\|S - V_1\|$  where  $V_1$  is one of the vertices and the estimate for the curvature  $\kappa$  of the underlying surface of the virtual original is given by the reciprocal value of the radius

$$\kappa = (\|S - V_1\|)^{-1}. \quad (3.2)$$

When the two triangles lie within the same plane, the radius of the sphere would be infinite and hence the curvature zero. Although this method yields an acceptable estimate for the curvature of a triangle pair, it cannot be applied to a single triangle. Moreover the above triangle-pair method is very likely to be sensitive to errors in data points. On the other hand it has inspired the subsequent approach.

A more appropriate method of curvature estimation takes the “smoothness” of the surface of a potential virtual original surface into account. Briefly the idea is as follows. The method allows nine different curvature estimates to be made for each triangle. Firstly, an “interpolated normal” at each vertex of a triangulated surface is calculated. Next for each triangle  $T$  a “compensated normal” is introduced that approximates better the surface normal of the virtual original.

Thus each triangle now has four normals associated (one for each vertex and one for the triangle).

When all the normals have been found a radius of curvature is estimated for each pair of normals. As we have four normals this method yields six pairs of normals and hence six curvature values (the inverses of the radii of curvature).

Additionally, for each of  $T$ 's adjacent triangles a radius of curvature is obtained in a similar manner by employing the compensated normals for that triangle pair. This yields a further three curvature values. Afterwards the resulting maximum and minimum of the nine values serve as principal curvature values  $\kappa_{\max}$  and  $\kappa_{\min}$  and the corresponding directions are taken as the directions of principal curvature. Because of the nine curvature values, the method is from now on referred to as **nine-fold evaluation of normals**, or short **NEN**. Finally mean and Gaussian curvature values can be calculated. In what follows the derivation of the method is described in more detail.

Section 3.2.1 describes how to determine interpolated normals at vertices of a triangulated surface, and how to establish a "compensated normal" for each triangle. The calculation of curvature values for each pair of normals with their associated points is explained in Section 3.2.2, where also the directions of principal curvature are estimated.

### 3.2.1 Defining interpolated and compensated normals

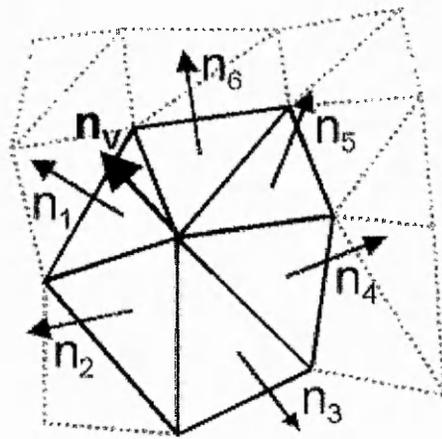
Consider a vertex  $V$  joined to a set of surface triangles  $T_i$  with associated unit normals  $\mathbf{n}_i$ . Then for  $V$  the **interpolated normal**  $\mathbf{n}_V$  is defined as

$$\mathbf{n}_V = (\sum w_i \mathbf{n}_i) / \|\sum w_i \mathbf{n}_i\| \quad (3.3)$$

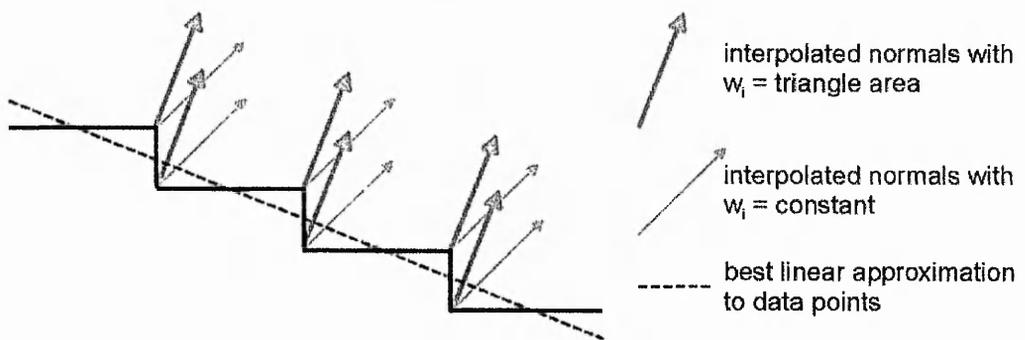
in which  $w_i$  are suitable weights. The idea of weighted average normals is used in [Hoschek et al. 98] for the determination of feature lines obtained from "angular variation". [Choi 91] (pp. 184) employs for each triangle sharing  $V$  the distance from  $V$  to the bisection point of its opposite edge as a weight. This is particularly unfavourable for long, thin triangles, because small errors can tilt the normal and can thus cause big deviations in the true surface normal of the virtual original.

For this reason the weights  $w_i$  are set to the areas of the triangles. Figure 3.2.1

illustrates how the interpolated normal  $\mathbf{n}_V$  depends on the triangle normals  $\mathbf{n}_i$ . The interpolated normal attempts to emulate the normal of the virtual original at the data point  $V$ . Figure 3.2.2, in cross section, clarifies the effect for different choices of weights for a particular example. Selecting equal weights for all normals (for the example  $w_i = 1$  the normals are shown as light grey vectors in Figure 3.2.2) would deteriorate the emulation of the true surface normal whereas weighting with triangle areas improves the alignment of the normals (dark grey vectors in Figure 3.2.2).



**Figure 3.2.1:** Illustration of how the interpolated normal  $\mathbf{n}_V$  depends on the triangle normals  $\mathbf{n}_i$



**Figure 3.2.2:** Illustration of how the interpolated normals are affected by the weights  $w_i$

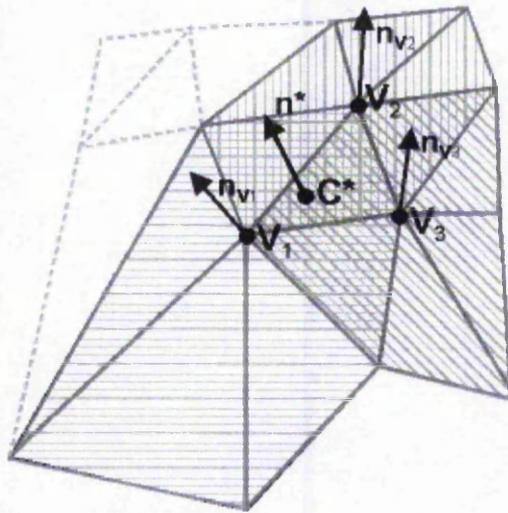
Furthermore for each triangle  $T = [V_1, V_2, V_3]$  a **compensated centre**  $C^*$  and a **compensated normal**  $\mathbf{n}^*$  are calculated by

$$C^* = (\sum u_j V_j) / (\sum u_j) \quad (3.4)$$

and

$$\mathbf{n}^* = (\sum u_j \mathbf{n}_{V_j}) / \left\| \sum u_j \mathbf{n}_{V_j} \right\| \quad (3.5)$$

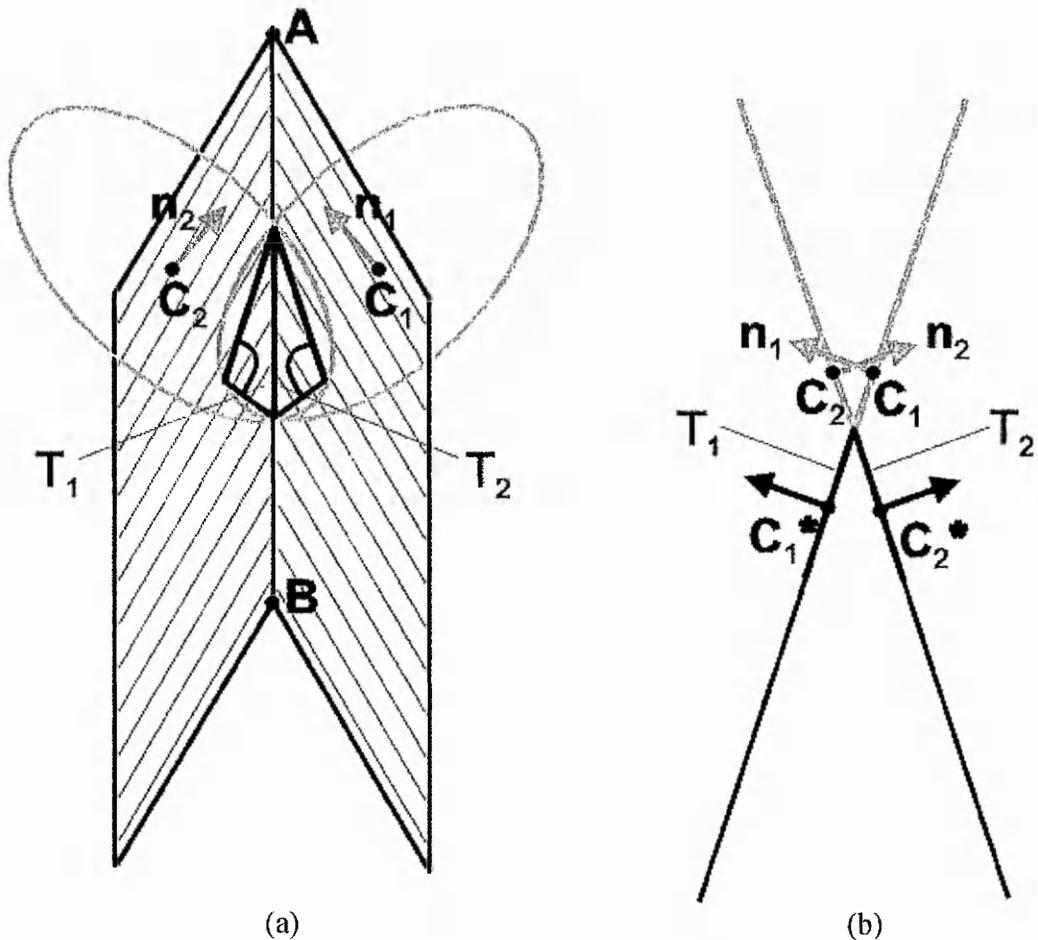
where weight  $u_j$  is the total area of triangles meeting in vertex  $V_j$  (for example, in Figure 3.2.1 the total area of triangles with black edges). Figure 3.2.3 illustrates the geometric meaning of the compensated centre  $C^*$  and the compensated normal  $\mathbf{n}^*$ , where each weight  $u_j$  is the area of the hatched area around its corresponding vertex  $V_j$ . An important consequence of the definition of the compensated centre of a triangle is that it is always within the triangle unlike the centre of a circumcircle.



**Figure 3.2.3:** *Geometric meaning of compensated centre  $C^*$  and the compensated normal  $\mathbf{n}^*$*

The motivation for introducing compensated centres is that the natural choice for an appropriate centre of a triangle (such as a centre of a circumcircle) does not take the structure of a triangulated surface into account. If a sphere is imagined, then the centre of the circumcircle through any three points on its surface would be located within the sphere. Thus one would expect that the centre of circumcircle is an appropriate basis for further geometric computations. However, this idea fails, even for data points without any noise. For example, consider the pair of triangles  $T_1$  and  $T_2$  in Figure 3.2.4 (a) each having an obtuse angle as indicated. These adjacent triangles share an edge that is placed on the join of two planes meeting each other in the line from **A** to **B** in an acute angle. Moreover, the triangles have the centres of circumcircles  $C_1$  and  $C_2$  that are located “above” the surface. In this case a convex surface (indicated by the black lines and black normals in Figure 3.2.4 (b)) may be misinterpreted as locally

concave when using the DN method of estimating curvature as it allows for a configuration with the normals  $\mathbf{n}_1$  and  $\mathbf{n}_2$  (grey arrows in Figure 3.2.4) passing through  $C_1$  and  $C_2$ . This happens because  $C_1$  and  $C_2$  are located outside the triangles by which they are defined. On the contrary employing the compensated centres  $C_1^*$  and  $C_2^*$  as well as the corresponding normals (in black) for the same task will not lead to a misinterpretation.



**Figure 3.2.4:** *An example where the choice for triangle centres  $C_1$  and  $C_2$  as centres of circumcircles can make a convex surface being found to be locally concave if the DN formula is applied*

- (a) *Two triangles  $T_1$  and  $T_2$  lying in two planes meeting in the line  $AB$ ; the centre  $C_1$  belongs to  $T_1$  and  $C_2$  belongs to  $T_2$*
- (b) *Cross section of (a) perpendicular to  $AB$  showing the better choice  $C_1^*$  and  $C_2^*$  as compensated centres*

As with the interpolated normals in Figure 3.2.2 a compensated normal attempts to achieve a better representation of the underlying surface.

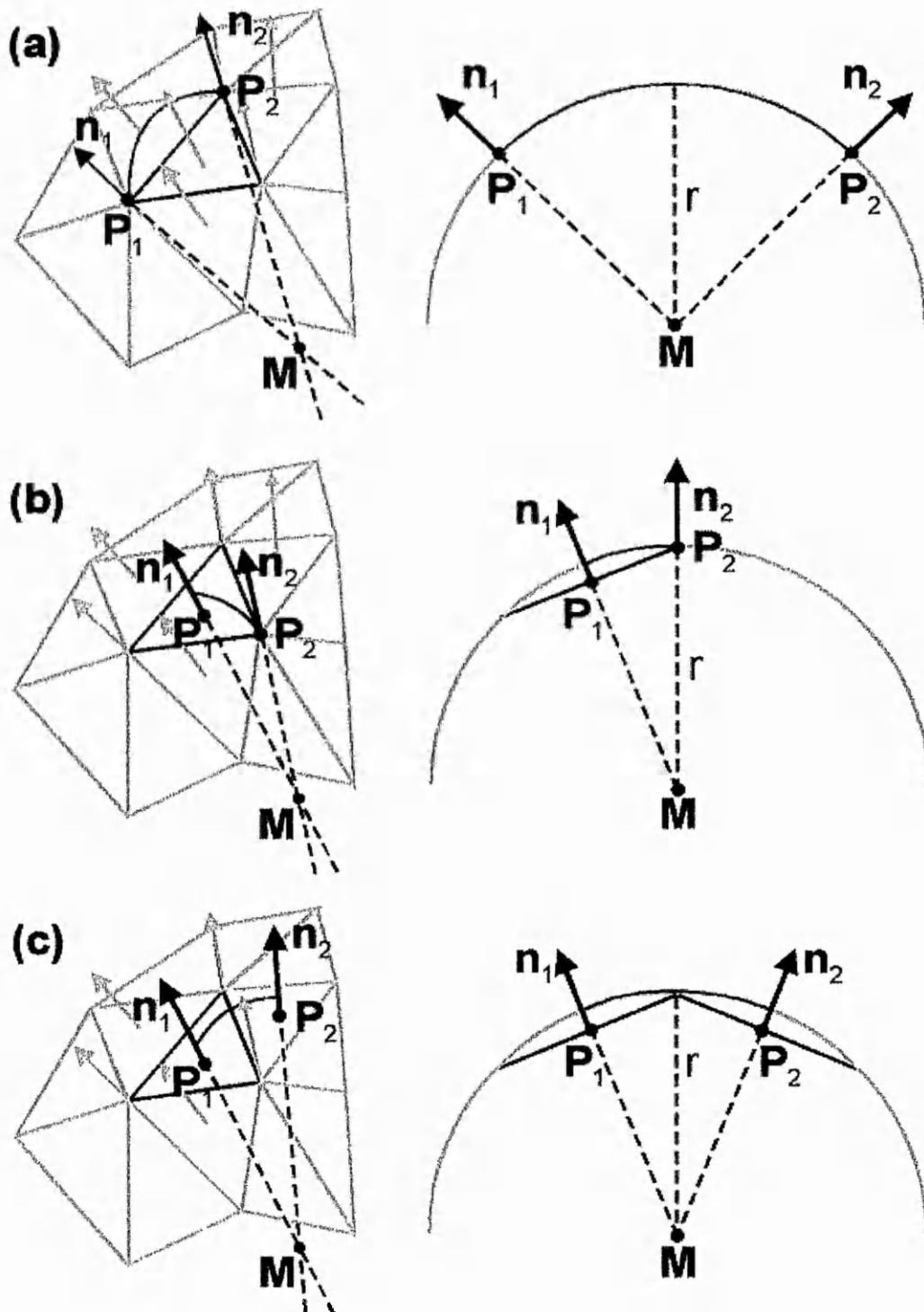
There are many possible ways other than the geometric concept of compensated centres and compensated normals that has been presented here. However, this concept has been found to produce satisfactory results when applied to curvature estimation on triangulated surfaces. It should be stressed that this concept does not rely on a parameterisation of the data.

At this stage interpolated normals with vertices and a compensated normal associated with a compensated centre are available for each triangle.

### **3.2.2 Estimation of principal curvature values and directions**

Next follows the method of estimating curvature values on triangulated surfaces using two points  $\mathbf{P}_1$  and  $\mathbf{P}_2$  and two associated normals  $\mathbf{n}_1$  and  $\mathbf{n}_2$ . In contrast to the requirements for using the DN curvature formula in Equation (3.1) the points are not necessarily vertices (i.e. data points). The direction for the curvature is the direction from  $\mathbf{P}_1$  to  $\mathbf{P}_2$ . Figure 3.2.5 illustrates the basic idea using the lines parallel to  $\mathbf{n}_1$  and  $\mathbf{n}_2$  (shown as hatched lines on the left-hand side) that pass through the points  $\mathbf{P}_1$  and  $\mathbf{P}_2$ , respectively, for three different configurations of these points and normals. The special configuration where these lines meet allows formulae to be derived involving only  $\mathbf{P}_1$ ,  $\mathbf{P}_2$ ,  $\mathbf{n}_1$ , and  $\mathbf{n}_2$  and can be applied even to those configurations in which the lines do not meet. If these lines intersect, the intersection point  $\mathbf{M}$  can be considered as the centre of a sphere through  $\mathbf{P}_1$  and  $\mathbf{P}_2$  on which  $\mathbf{n}_1$  and  $\mathbf{n}_2$  are perpendicular. Curvature can now be defined as the inverse of the radius of the sphere passing through  $\mathbf{P}_1$  and  $\mathbf{P}_2$  having the centre  $\mathbf{M}$  (Figure 3.2.5).

It is noted that Figures 3.2.5 and 3.2.6 represent 2D projections of effectively 3D configurations.



**Figure 3.2.5:** *The principle for estimating the curvature  $\kappa=1/r$  using two points  $P_1$  and  $P_2$  with associated normals  $n_1$  and  $n_2$  where*

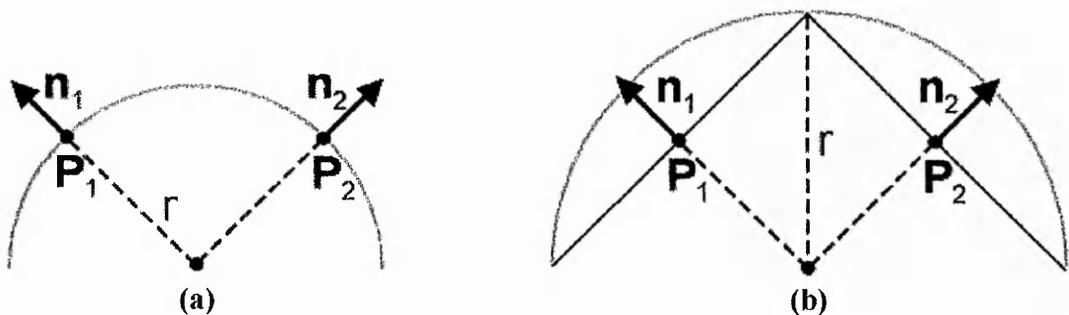
- (a)  $P_1$  and  $P_2$  are both on the surface (as used by [Ittner & Jain 85])
- (b)  $P_1$  is "inside" and  $P_2$  on the surface
- (c)  $P_1$  and  $P_2$  are both "inside" the surface

It is recalled that Equation (3.1) has presented the DN curvature formula for two points  $P_1$  and  $P_2$  on a surface such as in Figure 3.2.5 (a) and Figure 3.2.6 (a).

Geometrically this formula can be interpreted as the angular deviation of a pair of normals located at  $P_1$  and  $P_2$  on a surface scaled by the reciprocal distance between these points. However, when at least one of these points is not on the surface, a new more appropriate curvature formula has been found to be

$$\kappa = \|\mathbf{n}_1 \times \mathbf{n}_2\| / \|\mathbf{P}_1 - \mathbf{P}_2\| \quad (3.6)$$

(the configurations of points and normals in Figure 3.2.5 (b), (c) and Figure 3.2.6 (b)) which shall be denoted by **interior point (curvature) formula**. A justification for both curvature formulae is given in Appendix B.



**Figure 3.2.6:** Illustration of two different configurations of two points  $P_1$  and  $P_2$  with associated normals  $\mathbf{n}_1$  and  $\mathbf{n}_2$  for the estimation of curvature  $\kappa = 1/r$ . The same values for  $P_1$ ,  $P_2$ ,  $\mathbf{n}_1$ , and  $\mathbf{n}_2$  yield different values of curvature.

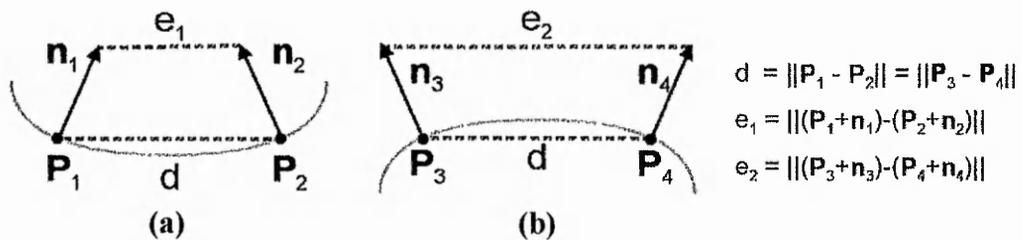
- (a) Configuration used by [Ittner & Jain 85] where  $P_1$  and  $P_2$  are on the surface
- (b) Configuration for  $P_1$  and  $P_2$  "inside" the surface; the curvature in this case is smaller, since the radius is larger

In general for the configurations in Figure 3.2.5 and 3.2.6 the lines parallel to  $\mathbf{n}_1$  and  $\mathbf{n}_2$  passing through  $P_1$  and  $P_2$  do not necessarily intersect. However, an intermediate point  $M$  can be thought of the point being the closest to both lines.

Now for each triangle  $T = [V_1, V_2, V_3]$  of a triangulated surface the DN curvature formula is applied to each pair of its vertices in turn with their associated interpolated normals (see Figure 3.2.5 (a)). This yields three values  $\kappa_j$  as curvature estimates. Applying the interior point formula to the compensated centre  $C^*$  of  $T$  and its associated normal  $\mathbf{n}^*$  paired with each of the vertices  $V_j$  and its associated interpolated normal  $\mathbf{n}_j$  in turn (as shown in Figure 3.2.5 (b)) gives a further three estimated curvature values. Finally, applying the interior point formula to  $C^*$  and  $\mathbf{n}^*$  paired with each of the compensated centres of  $T$ 's

neighbours and their associated normals gives three more curvature values  $\kappa_j$  (Figure 3.2.5 (c)). Because of these nine estimated curvature values the method is called nine-fold evaluation of normals (NEN). The curvature formulae implicitly take the dimensions of the surface triangles into account inasmuch as the distance between any two points  $P_1$  and  $P_2$  depends directly on the size of  $T$  and its adjacent triangles.

So far curvature for a triangle  $T$  has been estimated only in magnitude. The idea is to provide the curvature values with a sign. Therefore suppose that the directions of the triangle normals are consistently oriented for the whole surface and this is called “outward”. Each curvature value  $\kappa_j$  is defined as negative when the estimated centre of the sphere through the vertices of  $T$  is “inside” the triangulated surface relative to the triangle normal, and positive otherwise. More concisely, the curvature sign is defined to be positive for concave surfaces and negative for convex ones (see Figure 3.2.7).



**Figure 3.2.7:** Usual definition of the curvature sign for two points  $P_i$  and  $P_j$  with corresponding surface normals  $n_i$  and  $n_j$   
 (a) Positive curvature ( $e_2 < d$ )  
 (b) Negative curvature ( $e_1 > d$ )

A more formal criterion is presented in [Flynn & Jain 89] (similar to [Hoffman & Jain 87]) defining  $\kappa_j$  as

$$\begin{aligned} &\text{positive, if } \|P_1 - P_2\| \leq \|(P_1 + n_1) - (P_2 + n_2)\|; \\ &\text{negative, otherwise.} \end{aligned} \tag{3.7}$$

Confusingly, this definition reverses the sign of curvature defined above and commonly used in the majority of literature (as can be seen directly in [Tanaka et al. 98] or be deduced, e.g. from the curvature sign maps in [Besl & Jain 88], [Yokoya & Levine 91], [Fischer 99] or from the inexact table 1 in [Hilton et al.

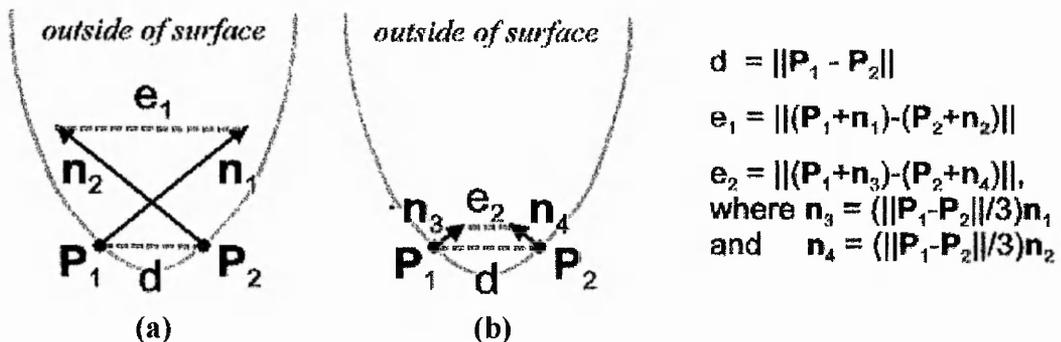
95], where the columns below “ $K>0$ ” and “ $K<0$ ” need to be swapped). However, by reversing the inequality it agrees with the usual result and is referred to as reversed Flynn & Jain criterion (for determining the sign of curvature).

There is, however, a more serious problem with this criterion: although it seems to be intuitively correct, it can in fact give the wrong sign when  $\|P_1 - P_2\| \leq 2$ . This means that the corresponding parts of two surfaces that are identical apart from scale will sometimes be given curvature of different sign. More precisely, if the surface is concave, then in some cases the term

$$\|(P_1 + n_1) - (P_2 + n_2)\| = \|n_1 - n_2 + P_1 - P_2\| \quad (3.8)$$

may exceed  $\|P_1 - P_2\|$  (see Figure 3.2.8 (a)). This would result in negative curvature in the sense of the reversed Flynn & Jain criterion. However, in this thesis negative curvature is associated with a convex surface.

The above formal criterion for the curvature sign can be remedied by re-scaling the normals to have length  $< \frac{1}{2}\|P_1 - P_2\|$ , e.g. by replacing each  $n_i$  by  $(\|P_1 - P_2\|/3) n_i$ . This replacement avoids the “crossing of normals” as is shown in Figure 3.2.8 (b).



**Figure 3.2.8:** Example of a concave surface where the reversed Flynn & Jain criterion for estimating the sign of curvature fails because

$$\|P_1 - P_2\| \leq 2 \text{ and } \|n_1\| = \|n_2\| = 1.$$

(a) The criterion used by [Flynn & Jain 89] applied to  $P_1$ ,  $P_2$ ,  $n_1$ , and  $n_2$

(b) The modified criterion developed by the Author employing  $P_1$ ,  $P_2$ ,  $n_3$  and  $n_4$

Thus after this revision the **sign of each curvature value**  $\kappa$  is defined as

$$\text{positive, if } \|P_1 - P_2\| \leq \|(P_1 + (\|P_1 - P_2\|/3) n_1) -$$

and  $(\mathbf{P}_2 + (\|\mathbf{P}_1 - \mathbf{P}_2\|/3) \mathbf{n}_2) \parallel$ ;  
 negative, otherwise. (3.9)

This definition is now independent of the scaling of the surface.

As for the magnitude of the curvature values, the above condition for the determination of their sign can be evaluated for all the nine curvature values.

Now the **principal curvature values** can be calculated as

and  $\kappa_{\max} = \max_j \kappa_j$   
 and  $\kappa_{\min} = \min_j \kappa_j$  (3.10)

with the correct sign.  $\kappa_{\max}$  and  $\kappa_{\min}$  allow a simple qualitative surface classification as described in Section 2.3.

In [Besl & Jain 88] the commonly used mean curvature

$$H = (\kappa_{\max} + \kappa_{\min}) / 2 \quad (3.11)$$

and the Gaussian curvature

$$K = \kappa_{\max} \kappa_{\min} \quad (3.12)$$

are introduced which also allow a surface classification into eight fundamental surface types or, for example, a classification, which is additionally based on torsion (such as in [Kehtarnavaz 88]).

Along with the principal curvature values for each triangle the principal curvature directions  $\delta_{\max}$  and  $\delta_{\min}$  can be estimated. They are taken to be the directions corresponding to the maximum and minimum values of principal curvature as introduced in Section 2.3.1. In practice each of these directions is determined from the two points used in the above estimation of principal curvature. These two points may be, for example, a compensated centre  $\mathbf{C}^*$  of a triangle and one of its vertices  $\mathbf{V}_j$ , so that the associated direction  $\delta$  would be either defined as  $\mathbf{C}^* - \mathbf{V}_j$  or as  $\mathbf{V}_j - \mathbf{C}^*$  (see remark in Section 2.3.1).

In the case of planes and spheres principal directions cannot be defined uniquely because all curvature values are equal, and thus no directions of preference exist.

Fortunately, the determination of initial parameters for plane and sphere does not require such directions. However, they will be used in the next chapter for the determination of the initial estimates for the axes of cylinders, cones and tori.

### 3.3 Appropriate synthetic and “real” test data

In order to demonstrate the effectiveness of the curvature estimation method, it has been applied to three selected sets of triangulated range data, one without noise and two with different levels of noise. Each of the three sets covers the primitive types plane, sphere and cylinder, so that the results can be compared to those in [Flynn & Jain 89]. However, in this paper the grid dimension required for a parametric description of the applied 2½D images is not revealed, i.e. no lower and upper bounds for  $u$  and  $v$  are given for the surface defined by  $z = f(u, v)$ . For data with added noise the scale-dependence of curvature affects the resulting values, and thus a fair comparison of estimation results of the DN with the DC and the NEN method has not been possible.

Each data set has been created as follows: on a regular grid with separation 0.005 between points in  $x$  and  $y$  of size  $128 \times 128$ ,  $z$ -values have been calculated where they fall on the surface of the primitive. Thus the number of points per image depends on the parameters of the equations. Each surface is represented by one of the following implicit equations:

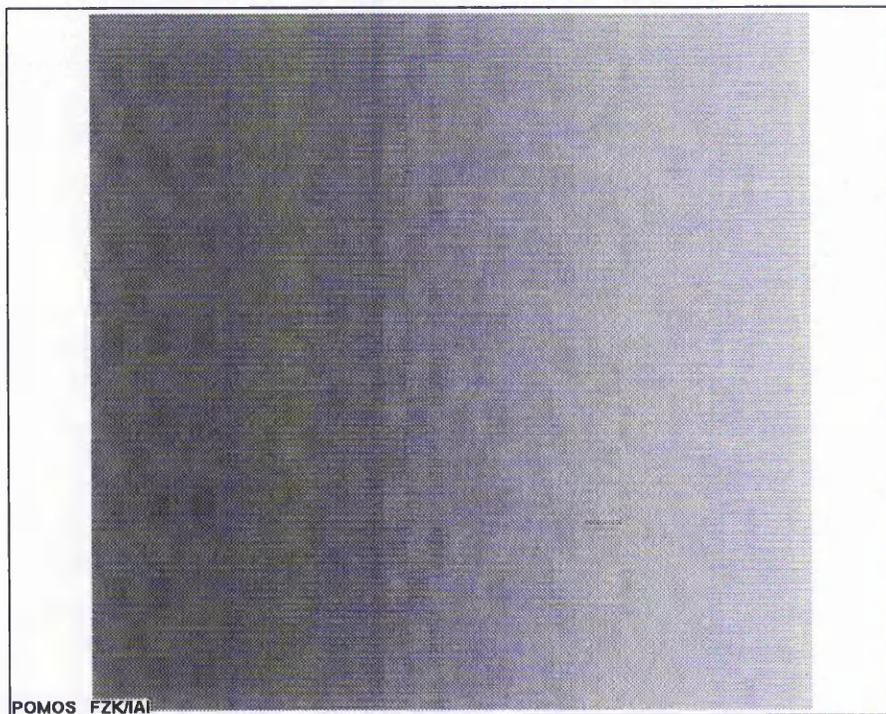
- I. **Plane:**  $f(x, y) = (1.0 - 0.1 x - 0.2 y) / 0.7$ .
- II. **Sphere:**  $f(x, y) = (x^2 + y^2 - r^2)^{1/2}$ .
- III. **Cylinder:**  $f(x, y) = (x^2 - r^2)^{1/2}$ .

The radius  $r$  in II. and III. has been varied with the values 1.0, 2.0, 5.0 and 10.0, the lowest  $x$ - and  $y$ -value for II. and III. are  $-3.2$ . For equation I. the lowest  $x$ - and  $y$ -values are 0.0. Each of the data points is represented as a 32-bit number (corresponding to the data type “float” in the programming language C for many computing systems).

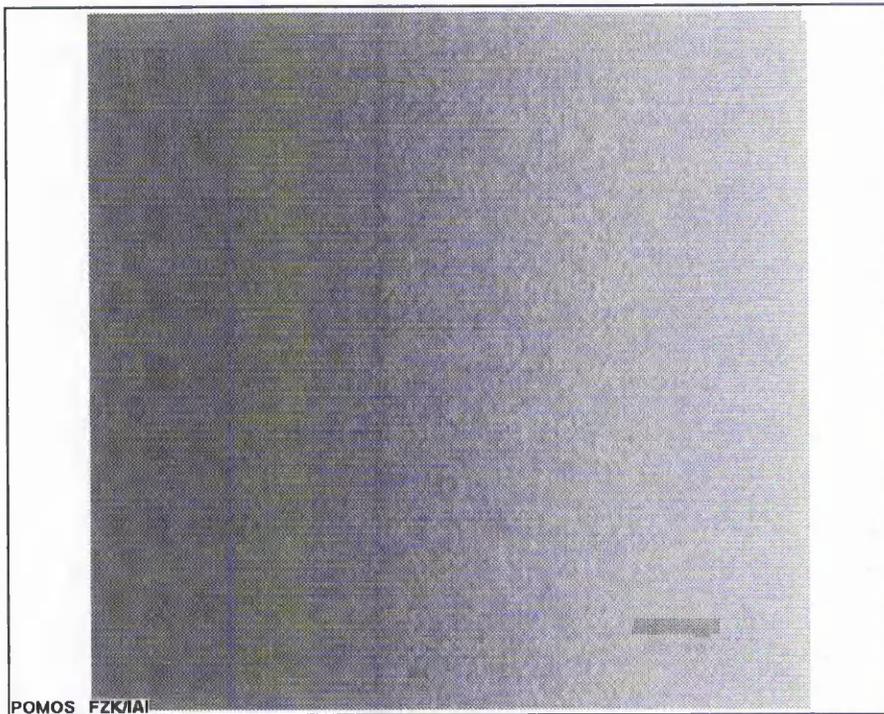
Set “A” of data points has been created without any noise in order to allow a comparison to the methods in [Flynn & Jain 89]. For set “B” each  $z$ -value  $f(x, y)$

in every equation has been increased by a random number in the interval  $[0.0, 0.0003]$ . For set "C" this interval has been enlarged to  $[0.0, 0.01]$  in order to produce a visibly non-smooth surface. Introducing such an element of uncertainty into the data aims to simulate measurement inaccuracies which occur when the data is obtained by a physical scanning system. Each data set has been converted to a triangulated surface using the CAX tool POMOS described in Section 1.5. Figures 3.3.1-3.3.3 give an impression of the surface smoothness for the cylinder of radius 10.0 from each data set.

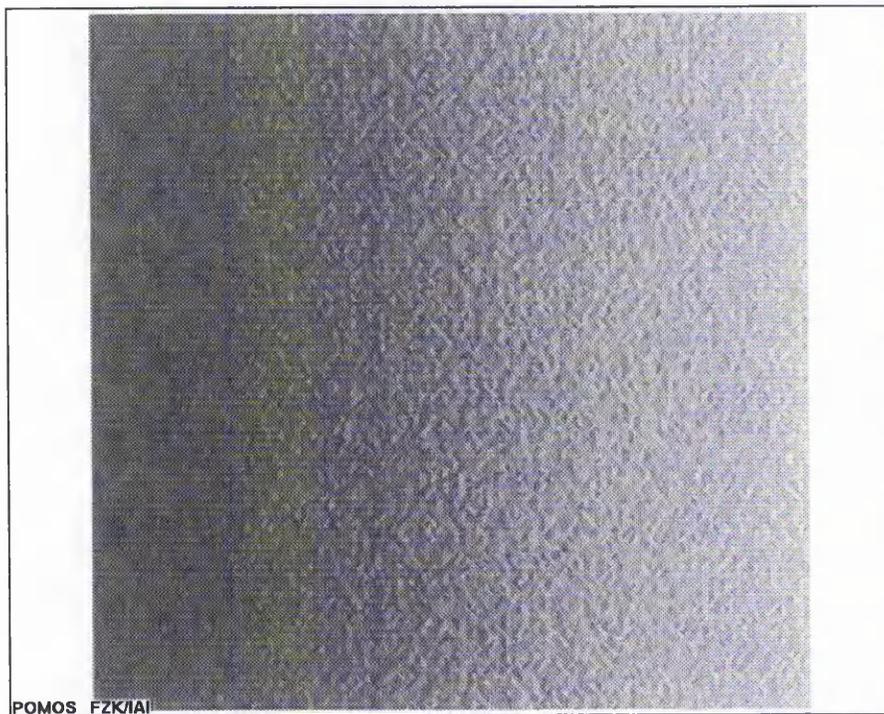
Unlike in [Flynn & Jain 89] the data has not been filtered or smoothed. This is because for serious reverse engineering applications such as design and construction (which is assumed in this project) the applied scanning device usually shows sufficiently high measurement accuracy.



**Figure 3.3.1:** *Surface sample of synthetic cylindrical data of radius 10.0 from data set "A" (without added "noise")*



**Figure 3.3.2:** *Surface sample of synthetic cylindrical data of radius 10.0 from data set "B" (with "noise" in the interval [0.0, 0.0003])*



**Figure 3.3.3:** *Surface samples of synthetic cylindrical data of radius 10.0 from data set "C" (with "noise" in the interval [0.0, 0.01])*

### 3.4 Comparison of curvature estimation methods

Principal curvature values  $\kappa_{\max}$  and  $\kappa_{\min}$  for the “discrete curvature” methods presented in Section 3.1 are listed for comparison, where these values have been computed using the curvature inversion formulae

$$\kappa_{\max} = H + (H^2 - K)^{1/2} \quad (3.2.12)$$

and

$$\kappa_{\min} = H - (H^2 - K)^{1/2}. \quad (3.2.13)$$

The overall test procedure is the following. Firstly, the DC, DN and NEN methods are applied to the data set “A” consisting of synthetic, noise-free data of one plane, four spheres and four cylinders of varying radii. For the DN method the principal curvature values are listed in [Flynn & Jain 89], so they have been compared with the DC and NEN results, as shown in Tables 3.1 - 3.3 (with magnitudes of the values only, and three significant figures).

Then the noisy data sets “B” and “C” have been employed to allow a comparison between the DC and the NEN method with respect to principal curvature values ( $\kappa_{\min}$  and  $\kappa_{\max}$ ) and in addition with the Gaussian (K) and mean (H) values. For each of the data sets a central square subset of points has been used rather than the entire surface in order to avoid problems with the discrete curvature methods at surface boundaries. Each square region consists of those points with x- and y-values strictly between -0.5 and 0.5 (i.e. these values have been omitted). The total area of a portion U for the DC and the NEN method has been determined as a weighted sum of areas of the individual triangles, where the weight has been set to 1.0 for an interior triangle, 0.5 for a non-corner triangle at the boundary and 0.25 for a triangle at the corner of a region. Results of the different curvature values are shown in Tables 3.4 - 3.6 for data set “B” and Tables 3.7 - 3.9 for data set “C”. Again if  $H^2 < K$  the curvature inversion formulae obviously fail to produce real valued numbers. This is indicated by a “\*” in the above tables.

For principal curvatures, all methods generally perform better when noise in the data is lower. For data set “A” (without noise) the NEN method offers generally fairly good results that are slightly better than both DC and DN. For data set “B” (with low noise) the NEN method performs better than DC when the radii of spheres and cylinders are smaller, whereas for larger radii the differences are less

significant. When the data is relatively noisy as in data set “C”, the quality of the results seems to depend on the surface type. For planar data all curvature values of the NEN method can be considered acceptable in contrast to DC. For spherical surfaces both methods produce not very accurate results. For cylindrical surfaces the minimum curvature values of the NEN method are clearly better than those of DC, whereas no method can be preferred concerning the maximum curvature. Surprisingly the estimated mean curvature produced by the NEN method seems to be fairly robust to noise for all surface types considered.

However, the DC method has the remarkable drawback that in some cases it does not offer real principal curvature values at all because of the way these values are obtained. The NEN method has been particularly designed for estimating principal curvature values on triangulated surfaces and gives reasonable results when the data has sufficiently good quality. Since for reverse engineering purposes this is generally the case, the NEN method has been selected for determining initial characteristic parameter values in Chapter 4.

True value	DC	DN	NEN
$\kappa_{\min} = 0$	*	6.59e-3	0.00e-4
$\kappa_{\max} = 0$	*	6.85e-3	0.00e-4

**Table 3.1:** *Estimated principal curvature values for noise-free planar data from set "A" ("\*" = unobtainable value)*

$\kappa_{\min}$			
True value	DC	DN	NEN
1.0	0.44	1.00	0.98
0.5	*	0.49	0.49
0.2	0.06	0.19	0.20
0.1	0.03	0.07	0.10

$\kappa_{\max}$			
True value	DC	DN	NEN
1.0	0.65	1.07	1.02
0.5	*	0.51	0.51
0.2	0.29	0.21	0.20
0.1	0.33	0.13	0.10

**Table 3.2:** *Estimated principal curvature values for noise-free spherical data from set "A" ("\*" = unobtainable value)*

$\kappa_{\min}$			
True value	DC	DN	NEN
0.0	0.05e-4	11.9e-4	0.00e-4
0.0	0.21e-4	17.1e-4	0.00e-4
0.0	0.00e-4	43.5e-4	0.00e-4
0.0	0.90e-4	96.2e-4	0.00e-4

$\kappa_{\max}$			
True value	DC	DN	NEN
1.0	0.33	1.01	1.00
0.5	0.11	0.51	0.50
0.2	0.31	0.21	0.20
0.1	0.19	0.13	0.10

**Table 3.3:** *Estimated principal curvature values for noise-free cylindrical data from set "A"*

True value	DC	NEN
$\kappa_{\min} = 0$	*	0.000
$\kappa_{\max} = 0$	*	0.000
$K = 0$	0.000	0.000
$H = 0$	-0.001	0.000

**Table 3.4:** *Estimated principal, Gaussian and mean curvature values for low noise planar data from set "B" ("\*" = unobtainable value)*

$\kappa_{\min}$			$\kappa_{\max}$		
True value	DC	NEN	True value	DC	NEN
-1.0	-0.658	-1.217	-1.0	-0.444	-0.792
-0.5	*	-0.750	-0.5	*	-0.266
-0.2	-0.327	-0.338	-0.2	-0.062	-0.060
-0.1	-0.325	-0.181	-0.1	-0.030	-0.023

K			H		
True value	DC	NEN	True value	DC	NEN
1.0	0.292	0.964	-1.0	-0.551	-1.004
0.25	0.044	0.200	-0.5	-0.175	-0.508
0.04	0.020	0.020	-0.2	-0.194	-0.199
0.01	0.010	0.042	-0.1	-0.178	-0.102

**Table 3.5:** *Estimated principal, Gaussian and mean curvature values for low noise spherical data from set "B"*

$\kappa_{\min}$			$\kappa_{\max}$		
True value	DC	NEN	True value	DC	NEN
-1.0	-0.329	-1.079	0.0	-0.001	0.129
-0.5	-0.112	-0.537	0.0	-0.002	0.056
-0.2	-0.308	-0.196	0.0	-0.002	-0.002
-0.1	-0.191	-0.099	0.0	0.000	0.000

K			H		
True value	DC	NEN	True value	DC	NEN
0.0	0.000	-0.139	-0.5	-0.165	-0.475
0.0	0.000	0.030	-0.25	-0.057	-0.241
0.0	0.001	0.000	-0.1	-0.155	-0.099
0.0	0.000	0.000	-0.05	-0.095	-0.050

**Table 3.6:** *Estimated principal, Gaussian and mean curvature values for low noise cylindrical data from set "B"*

True value	DC	NEN
$\kappa_{\min} = 0$	-0.150	-0.018
$\kappa_{\max} = 0$	0.105	-0.009
$K = 0$	-0.016	0.000
$H = 0$	-0.023	-0.014

**Table 3.7** *Estimated principal, Gaussian and mean curvature values for high noise planar data from set "C"*

$\kappa_{\min}$			$\kappa_{\max}$		
True value	DC	NEN	True value	DC	NEN
-1.0	-0.654	-1.588	-1.0	-0.446	-0.444
-0.5	-0.233	-0.887	-0.5	-0.096	-0.133
-0.2	-0.226	-0.416	-0.2	-0.150	-0.007
-0.1	*	-0.210	-0.1	*	-0.008

K			H		
True value	DC	NEN	True value	DC	NEN
1.0	0.292	0.705	-1.0	-0.550	-1.016
0.25	0.022	0.118	-0.5	-0.164	-0.510
0.04	0.034	-0.003	-0.2	-0.188	-0.204
0.01	0.032	0.002	-0.1	-0.168	-0.109

**Table 3.8** *Estimated principal, Gaussian and mean curvature values for high noise spherical data from set "C" ("\*" = unobtainable value)*

$\kappa_{\min}$			$\kappa_{\max}$		
True value	DC	NEN	True value	DC	NEN
-1.0	-0.330	-1.132	0.0	0.000	0.153
-0.5	-0.106	-0.482	0.0	-0.012	-0.026
-0.2	-0.252	-0.219	0.0	-0.041	-0.010
-0.1	*	-0.146	0.0	*	-0.032

K			H		
True value	DC	NEN	True value	DC	NEN
0.0	0.000	-0.173	-0.5	-0.165	-0.489
0.0	0.001	0.012	-0.25	-0.059	-0.254
0.0	0.010	-0.002	-0.1	-0.146	-0.104
0.0	0.010	-0.005	-0.05	-0.080	-0.057

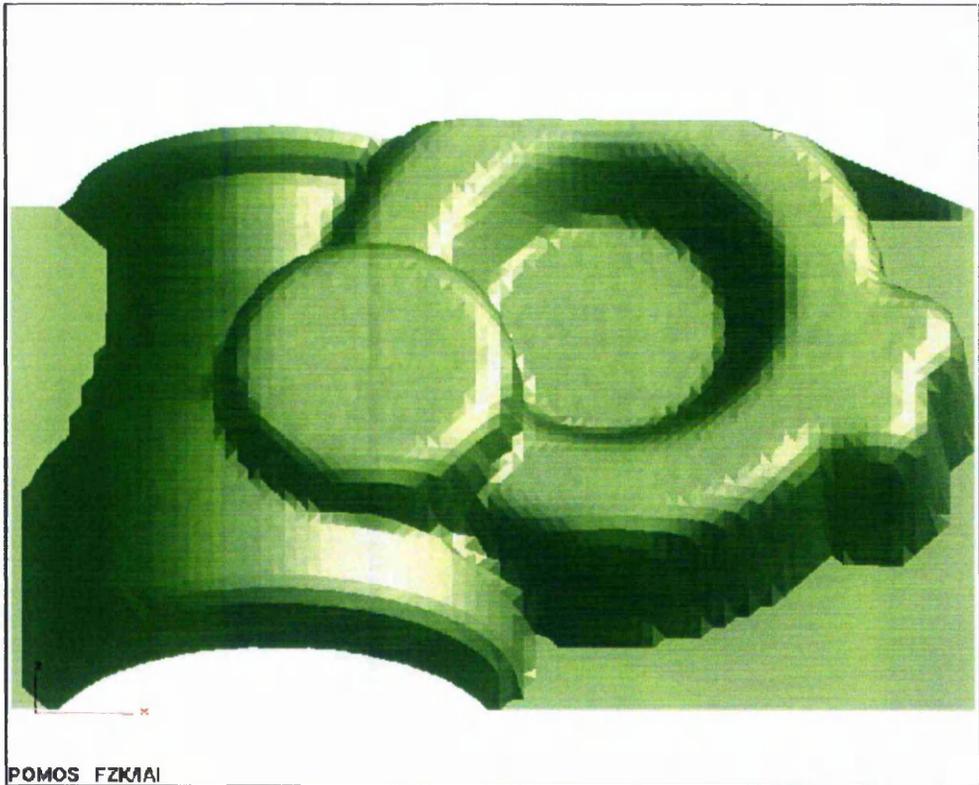
**Table 3.9** *Estimated principal, Gaussian and mean curvature values for high noise cylindrical data from set "C" ("\*" = unobtainable value)*

### **3.5 Application of the NEN method to non-synthetic data**

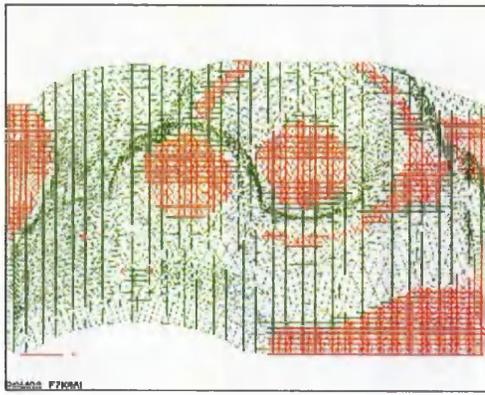
This section presents two examples of non-synthetic surfaces to illustrate estimated mean curvature values obtained from the application of the NEN method. The first example in Figure 3.5.1 shows the triangulated surface of a mechanical part (presumably acquired from a scan of the engine of a car; see [Hoschek 96] for the origin of this data set). The second example in Figure 3.5.3 presents the triangulated surface of a tooth of unknown origin. Each of the Figures 3.5.2 and 3.5.4 shows the same surface, respectively, in a wire-frame representation for different ranges of estimated mean curvature. All those surface triangles for which the magnitude of the estimated mean curvature is within the corresponding range are marked by red edges.

A visual examination of the results reveals a good correspondence of obviously planar regions with regions of estimated low mean curvature, in particular in Figure 3.5.2 (a). Furthermore it can be observed that all triangles of the cylindrical and cone-shaped regions on the left-hand side of Figure 3.5.2 (b) are within a small curvature range. This gives rise to the expectation that curvature estimation can be used to detect instances of these primitives in a triangulated surface. This could be achieved, for example, by clustering the estimated (mean) curvature values prior to a search for all those triangles whose curvature is in an appropriate range. For the detection of spheres it is expected that the difference between the estimated minimum and maximum curvature values for each triangle in a spherical region is small.

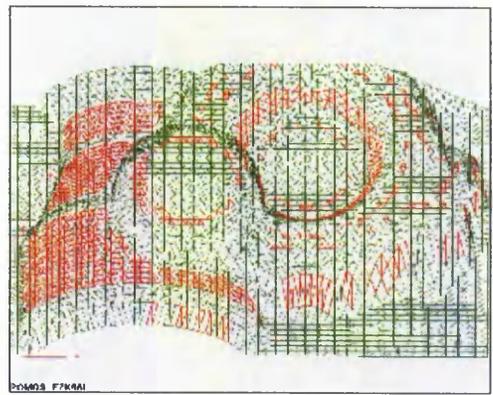
In contrast to the first example the surface of the tooth in Figure 3.5.3 does not allow for an obvious surface segmentation. However, as suggested above it can be attempted to detect instances of spheres, cylinders, and cones in this surface by using the estimated curvatures.



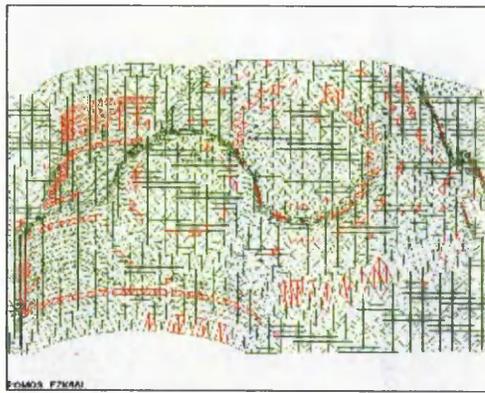
**Figure 3.5.1:** *Triangulated surface of a mechanical part (x-y-dimensions: approx. 81mm × 43mm; 4018 data points; 7776 triangles)*



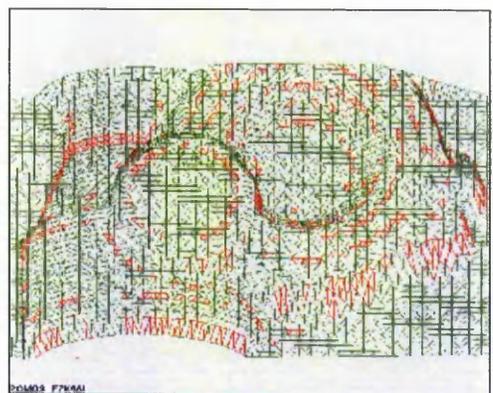
(a) 0.000 - 0.002



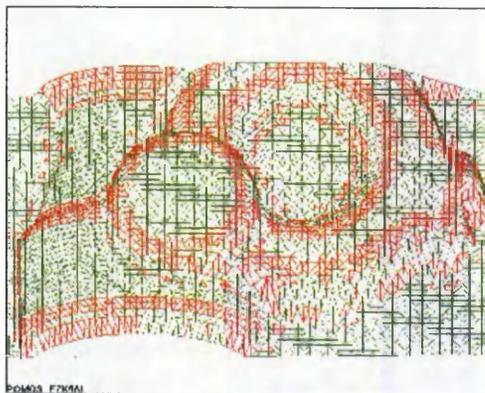
(b) 0.002 - 0.004



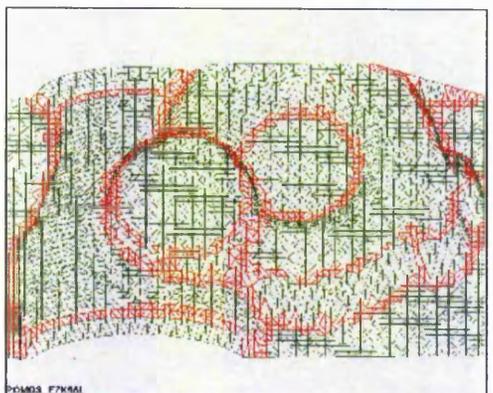
(c) 0.004 - 0.02



(d) 0.02 - 0.05

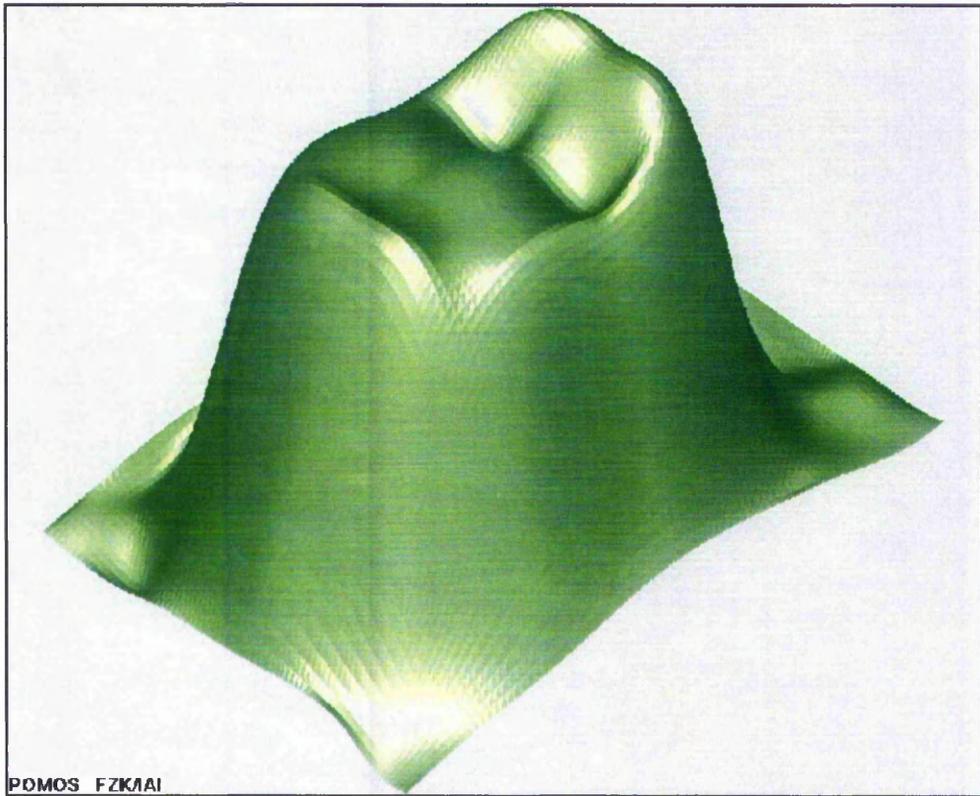


(e) 0.05 - 0.2



(f) 0.2 - 100.0

**Figure 3.5.2:** Mean curvature ranges for the triangulated surface in Figure 3.5.1



**Figure 3.5.3:** *Triangulated surface of a tooth (x-y-dimensions: approx. 11mm × 7.5mm; 6927 data points; 12800 triangles)*



(a) 0.000 - 0.002



(b) 0.002 - 0.004



(c) 0.004 - 0.02



(d) 0.02 - 0.04



(e) 0.04 - 0.2



(f) 0.2 - 100.0

**Figure 3.5.4:** Mean curvature ranges of the tooth model in Figure 3.5.3

### 3.6 Summary

Methods of curvature estimation for triangulated surfaces have been developed that attempt to compensate for errors in coordinates of vertices. This method is based on a two-stage averaging process. At first for each vertex of the surface an interpolated normal vector is computed (using a patch of a triangulated surface that consists of all triangles sharing the vertex). Secondly for each triangle on the surface a compensated normal is computed as the sum of the interpolated normals at its vertices where each vertex is weighted with the area of its corresponding patch. Afterwards for each triangle on the surface curvature values can be obtained employing the differences between 9 pairs of normals associated with the triangle and its neighbours. The maximum and the minimum of these values serve as the principal maximum and minimum curvature. In addition this method provides estimates for the principal curvature directions for each triangle, which is useful for subsequent surface analysis.

The method has been compared to existing methods suitable for estimating curvature on triangulated surfaces. For data of sufficiently good quality that is produced by scanning devices currently used for reverse engineering purposes the method shows satisfactory results. It is less good compared to the discrete curvature method for more noisy data, but in fact it still offers real valued results for principal curvature where the discrete method sometimes fails to give a result. Unlike the discrete method, the presented novel method can also be applied to estimate curvature at the border of a triangulated surface. Moreover, the capability to produce estimates for the principal curvature directions is a feature that makes the novel method superior to the discrete method.

Satisfactory results of curvature estimation have been shown for examples with synthetic noise-free, synthetic noisy and real data.

## **4 Region growing for geometric primitives on a triangulated surface**

For the segmentation of a triangulated surface in this thesis the following is relevant. The segmentation problem needs to be understood as tolerance-dependent because this has been tacitly assumed in the existing literature without explicit inclusion in the problem setting. Therefore, the segmentation problem given in Section 2.2.1 is reformulated more adequately in Section 4.1. Such a problem definition helps to evaluate the quality of the output that a segmenter (i.e. the implementation of a segmentation algorithm on a computer) produces for an input triangulated surface.

Moreover, it is necessary to cope with the theoretical problem on a more detailed and practical level, namely by applying the region growing method introduced in Section 2.4. A modified algorithm for region growing employed for this work is described in Section 4.2. Region growing encompasses two phases:

- a. Seed region identification, and
- b. Iterated region growth.

As pointed out in Section 2.5 the individual problems to solve are:

- a. Estimating the characteristic parameters from seed regions,
- b. Fitting an appropriate candidate surface in a growth step, and
- c. Identifying the most appropriate surface segment to extract after region growing.

In order to identify appropriate seed regions for each geometric primitive under consideration, the estimated principal curvature of the previous chapter guides the seed selection process. Triangles of estimated “low” principal curvature values are picked as seeds for planar segments. Suitable seeds for spheres require “high” principal curvature values of almost the same magnitude and the same sign, whereas for cones and cylinders one of the absolute values needs to be small while the other is notably larger in magnitude. Almost every triangle with “high” principal curvature can serve as a seed for a toroidal segment, because a full torus can be approximated locally by each of the fundamental surface types shown in Table 2.3.2.1.

By examining the design of modern surfaces it can be observed that a majority of them are composed merely of a relatively small number of “simple” surface parts without handles or holes. Therefore it seems reasonable to extract the geometric primitives of simpler shapes first (as in [Besl & Jain 88]). For planes, spheres, and cylinders a single seed triangle suffices for the determination of the initial parameters characterising each geometric primitive. On the contrary, a pair of adjacent triangles with “high” principal curvature is required for the estimation of initial parameters for cone and torus. A method for an approximate determination of these parameters is explained in Section 4.3.

In a further step each identified seed is grown successively to a segment of a geometric primitive with a maximum number of triangles. During the growing phase the initial characteristic parameters may likely to be adjusted to fit to each data point in the current region and to those that are added. This adjustment generally involves numerical optimisation and is performed by a Genetic Algorithm (GA) followed by a “Direct Search” method as is explained in more detail in Section 4.4. However, for the extraction of planar segments these parameters can be readjusted by a quick geometric method not involving numerical optimisation, as described in Chapter 5. This is particularly useful since planar segments occur often in technical surface designs.

Section 4.5 discusses the various algorithmic parameters that are used for entire region growing process. For example, a parameter referred to as CURVATURE controls the discrimination between “low” and “high” curvature values. As a second example, the tolerance parameter  $\tau$  rules over the “goodness of fit” of the segment that approximates an extracted region. Algorithmic parameters for the control of the GA are also explained in this section.

Examples of identified segments on triangulated surfaces are presented in Section 4.6. Both synthetic and noisy data samples are shown with the characteristic parameters obtained from the surface fitting process. It shall be noted that each extracted segment is a (regular) triangulated surface in its own right. Section 4.7 discusses the modified region growing algorithm, the methods of estimating characteristic parameters of geometric primitives presented in Section 4.3, and

the segmentation results of Section 4.6. Finally, the chapter is summarised in Section 4.8.

#### 4.1 Problem definition in the context of region growing

Firstly, the triangulated surfaces are demanded to be “regular” in order to avoid pathological cases. Next, the term region - previously only used for range images - is defined for a triangulated surface.

Let  $S = (V, E, T)$  be a triangulated surface consisting of a set  $T = \{T_1, \dots, T_m\}$  of triangles, a set  $V = \{V_1, \dots, V_k\}$  of vertices in  $T$ , and a set  $E = \{(V_i, V_j) \subseteq T: V_i, V_j \in V \text{ for at least one } T \in T, \text{ where } i \neq j\}$  of edges.  $S$  is said to be **regular**, if

- a. each triangle in  $T$  has positive area;
- b. the neighbourhood of each non-border point  $P$  on  $S$  is **locally parameterisable** (i.e. the neighbourhood of  $P$  can be mapped bijectively onto a bounded, connected subset of a 2D plane “without holes”);
- c. for every combination of two triangles  $T_i, T_j$  in  $T$  there exists a sequence of edge-connected triangles that starts with  $T_i$  and ends with  $T_j$ .

Let  $S$  be a regular triangulated surface. The non-empty set  $R = (V_R, E_R, T_R)$  is said to be a **region** (of  $S$ ), if  $T_R \subseteq T$ ,  $E_R$  is the set of all triangle edges in  $T_R$ ,  $V_R$  is the set of all triangle vertices in  $T_R$ , and  $R$  is regular.

Let  $S$  be a triangulated surface, and a tolerance  $\tau > 0$ . Roughly the **segmentation problem** consists of determining a minimal number  $n(\tau) > 0$  and a decomposition of  $S$  into disjoint regions  $S_1, \dots, S_{n(\tau)}$  such that

- a. their union yields  $S$ ;
- b. each  $S_i$  is of “homogeneous shape” within tolerance  $\tau$  and connected.

A triangulated surface  $S = (V, E, T)$  (or a part of it) is of “homogeneous shape” within tolerance  $\tau$  if it can be approximated by a geometric primitive  $g \in G$  such that all points of  $S$  have a distance to the surface of  $g$  that is smaller than  $\tau$ . This can be expressed by employing the implicit functions of the geometric primitives which model the point-to-surface distance as follows:

$$\|f_g(\mathbf{P}; \mathbf{X}_g)\| < \tau \quad \text{for all } \mathbf{P} \in \mathbf{V} \quad (4.1)$$

where  $\mathbf{X}_g$  denotes a fixed parameter vector characterising position, orientation, and size of  $g$ .

## 4.2 The modified region growing algorithm for triangulated surfaces

An algorithm for segmenting a triangulated surface into parts of geometric primitives has been established by the Author and is presented in this section. It is mainly based on the region growing approach described in Section 2.4, formulated in Figure 4.2.1, and its detailed explanation follows next. Here  $\mathbf{S}$  is assumed to be a (regular) triangulated surface, and  $\tau > 0$  a small tolerance value.

In step 1 a local measure of curvature is estimated for each triangle. Each one is classified as having either low or high curvature depending on an algorithmic parameter CURVATURE that is adjusted empirically. Details are given in Section 4.5.

Step 2 forms a nested loop where for each primitive type a region growing scheme is performed. For the remainder of this paragraph an arbitrary type of geometric primitive  $g \in \mathbf{G}$  is considered as fix. In phase A each (initially untagged) triangle  $T$  receives a tag  $\text{TagOfGeoPrim}(g)$ , a tag that indicates the membership of  $T$  to a specific instance of the geometric primitive  $g$ , during an attempt to grow triangles to a homogeneous region. A subsequent evaluation decides about the success of this attempt. If it is successful, the tag for  $T$  is retained. Otherwise the tag  $\text{TagOfGeoPrim}(g)$  is replaced by the tag DISCARDED which means that  $T$  is not considered a second time as a seed triangle for growing a geometric primitive of type  $g$ . In this way information is being recorded which triangles have already been examined.

Phase B removes the tag DISCARDED from all triangles to provide untagged ones as seeds for the next iteration in step 2 performed on another type  $g \in \mathbf{G}$ . This method avoids ambiguities in the mapping of triangles to extracted surface segments: only discarded triangles are considered for region growing of other

primitive types since those triangles that are already assigned to a valid segment of a geometric primitive remain unchanged.

In more detail, the tagging phase A is composed of five sub-phases:

- a. Seed region selection,
- b. Initial tagging,
- c. Initial estimation of characteristic parameters,
- d. The actual region growing, and
- e. Region validation.

In sub-phase a triangles of estimated low mean curvature is selected as a seed region if the geometric primitive is of type PLANE, whereas for the other primitive types triangles of estimated high mean curvature are chosen. If there are no further possible seed regions, then the loop in step 2 will proceed with the next type of geometric primitive. Otherwise the loop is repeated for the same type of geometric primitive (but involves in general different characteristic parameters) until no further seed regions can be selected. The seed region search is exhausted if all initially untagged or not discarded triangles obtain the tag DISCARDED or the tag TagOfGeoPrim( $g$ ).

Once the seed triangles have been selected they are tagged with TagOfGeoPrim( $g$ ) in sub-phase b. The geometric primitives PLANE, SPHERE, and CYLINDER require one seed triangle only, whereas CONE and TORUS require two seed triangles (see Section 4.3 for details). Sub-phase c provides initial characteristic parameters for the seed region of the currently processed instance of  $g \in \mathbf{G}$ . These parameters are likely to require an adjustment during region growing.

Now in sub-phase d another loop starts: all triangles  $T$  adjacent to triangles of the current region  $R$  are potential candidates to enlarge it. Initially, these triangles are untagged; later on they also receive either the tag TagOfGeoPrim( $g$ ) or DISCARDED. For the actual region growing it can be observed that each non-seed triangle  $T$  in the current region  $R$  possesses one edge and hence two vertices in  $R$ . So in sub-phase d.i merely the “outer” vertex  $V$  (i.e. the vertex not belonging to  $R$ ) of  $T$  needs to be examined for the decision whether to include  $T$  in  $R$  or not.

This decision is made as follows. First, in sub-phase d.ii  $R$  and its corresponding characteristic parameters  $\mathbf{X}_g$  are saved in case the parameter adjustment fails. Next, in sub-phase d.iii  $T$  is attached to  $R$  “on trial” and the distance criterion is evaluated in sub-phase d.iv. If the result is positive, then the characteristic parameters describe a part of geometric primitive fitting to the enlarged region. A negative result necessitates an attempt to adjust the parameters in  $\mathbf{X}_g$  that is undertaken in the “else”-branch of sub-phase d.iv. Here, the distance criterion needs to be evaluated for *all* vertices in  $R$ . This is necessary because the attempt to adjust  $\mathbf{X}_g$  may imply a change of position, orientation, and scale of the surface part that is extracted as an instance of  $g \in G$ . Consequently, the distance of some vertices to this surface could exceed the tolerance they have met before. If this does not happen, the vertex  $V$  is included in  $R$  as before, otherwise the triangle  $T$  is tagged as DISCARDED, and the previous characteristic parameters for  $\mathbf{X}_g$  and region are restored.

In [Roth & Levine 93] it is stated that such an “extraction” of geometric primitives can be viewed as an optimisation problem where the goal is to find the global minimum of a so-called “cost function”. Such a function has potentially many local minima. This is true in the present case where the cost function corresponds to the distance function  $|f_g(V; \mathbf{X}_g)|$  in the “if”-clauses in sub-phase d.iv. The problem of finding the global minimum for  $\mathbf{X}_g$  is subject of Section 4.4.

In the post-processing sub-phase e the size of the region needs to be validated. If the number of triangles in  $R$  exceeds an algorithmic parameter `MIN_TRIANGLES` (defined in Section 4.5), then the boundary of each region  $R$  will be obtained as the set of all those edges not being shared by an adjacent triangle in  $R$ . It should be noted that the resulting boundary may consist of several isolated curves. Otherwise the seed triangles of  $R$  are tagged as DISCARDED to prevent them from being selected again for the current type of geometric primitive.

1. For all triangles  $T$  in  $S$  do:
  - Estimate curvature of  $T$
  - Classify  $T$  as having either LOW or HIGH curvature
2. For each geometric primitive  $g \in G$  do
  - A. While there are triangles in  $S$  that do not have any of the tags DISCARDED and TagOfGeoPrim( $g'$ ) for all  $g' \in G$  do
    - a. If  $g = \text{PLANE}$ :
      - Select untagged seed triangle for  $R$  having LOW curvature
      - Else
      - Select seed region  $R$  as 1 or 2 untagged triangles having HIGH curvature
    - b. Assign tag TagOfGeoPrim( $g$ ) to all triangles of  $R$
    - c. Determine initial estimate of  $X_g$
    - d. For all triangles  $T$  adjacent to triangles in  $R$  having no tag TagOfGeoPrim( $g'$ ) for **any**  $g' \in G$  do
      - i. Determine vertex  $V$  of  $T$  not already belonging to  $R$
      - ii. Store  $X_g$  as  $X_{\text{save}}$ ,  $R$  as  $R_{\text{save}}$
      - iii. Add  $T$  to  $R$
      - iv. If  $|f_g(V; X_g)| \leq \tau$ 
        - Assign tag TagOfGeoPrim( $g$ ) to  $T$
        - Else
          - I. Try to adjust parameter vector  $X_g$
          - II. If  $|f_g(V'; X_g)| \leq \tau$  for **all**  $V'$  in  $R$ 
            - Assign tag TagOfGeoPrim( $g$ ) to  $T$
            - Else
              - Restore  $X_g$  from  $X_{\text{save}}$ ,  $R$  from  $R_{\text{save}}$
              - Assign tag DISCARDED to  $T$
    - e. If number of triangles in  $R > \text{MIN\_TRIANGLES}$ 
      - Determine boundaries of  $R$
      - Assign  $X_g$  to  $R$
      - Else
      - Remove tag TagOfGeoPrim( $g$ ) from all triangles in  $R$
      - Assign tag DISCARDED to all triangles of the seed region
  - B. Remove tag DISCARDED from all  $T$  in  $S$  not having TagOfGeoPrim( $g'$ ) for **any**  $g' \in G$

**Figure 4.2.1:** *A modified region growing algorithm in pseudo-code to segment a triangulated surface into parts of geometric primitives*

### 4.3 The estimation of characteristic parameters from seed triangles

It is recalled that the modified region growing algorithm for surface fitting and extraction presented in Section 4.2 needs to accomplish two important steps besides the selection of seed regions: providing estimates for the characteristic parameters for the seed region, and adjusting these parameters to the region in a growth step every time a triangle is added. Thus this section is dedicated to the estimation of characteristic parameters for the different types of geometric primitives from seed regions. Apart from the extraction of planes, where the seed triangles estimated are presupposed to have “low” mean curvature, seed triangles for the other primitive types are supposed to have estimated “high” mean curvature.

Furthermore, the benefit of principal curvature directions as introduced in Section 2.3.1 and computed for triangulated surfaces in Section 3.2.2 is discussed briefly here. Except for planes and spheres where they are ambiguous these directions can be used to support the determination of characteristic parameters for geometric primitives. For example, on the surface of a convex cylinder the direction  $\delta_{\max}$  of maximum curvature (the reader is reminded that the maximum curvature itself is 0) is parallel to the axis  $\mathbf{a}_{\text{cyl}}$  of the cylinder, and thus an estimate for  $\mathbf{a}_{\text{cyl}}$  can be expressed in terms of  $\delta_{\max}$  only (see Section 4.3.3 for details). In general the usage of the principal directions needs to be handled with care since certain parts of a torus may be locally approximated by a sphere, and hence ambiguities for this type of geometric primitive can occur as for the sphere.

However, in order to obtain estimates of characteristic parameters for planes and spheres these directions are not required. If they are almost identical as in case of the torus the corresponding part of the triangulated surface is simply “skipped” in the seed selection step in 2.A.a of Figure 4.2.1 and it is attempted to grow a toroidal segment somewhere else.

In order to simplify the notation throughout the remainder of this thesis,  $\kappa_{\max}$  ( $\kappa_{\min}$ ) refers to the principal curvature value that is larger (smaller) in magnitude,

respectively. Analogously, the directions  $\delta_{\max}$  and  $\delta_{\min}$  of principal curvature are employed. Moreover,  $R_{\max}$ , the radius of maximum principal curvature, is denoted by  $1/\kappa_{\max}$ , whereas  $R_{\min}$ , the radius of minimum principal curvature, refers to  $1/\kappa_{\min}$  (although  $\kappa_{\max} > \kappa_{\min} > 0$  implies  $R_{\max} = 1/\kappa_{\max} < 1/\kappa_{\min} = R_{\min}$ ). A lower index  $i$  (such as in  $\kappa_{i, \max}$ ) indicates that the principal curvature value is associated with a triangle  $T_i$ , and similarly for principal directions and radii.

### 4.3.1 Plane

The implicit equation of a plane can be given in different forms. In intercept form

$$x/a + y/b + z/c = 1 \quad (4.2)$$

the plane passes through three points  $(a, 0, 0)$ ,  $(0, b, 0)$ , and  $(0, 0, c)$ . Another form to describe a plane may involve spherical coordinates  $(r, \phi, \theta)$ . These coordinates can represent any position of a plane in 3D space after applying three operations on the initial plane  $z = 0$ . The first operation shifts this plane by the distance  $r \geq 0$  up along the  $z$ -axis, and the following two operations rotate the resulting plane around the  $y$ - and the  $x$ -axis (by the angles  $\phi \in [0, \pi]$  and  $\theta \in [0, 2\pi)$ , respectively).

However, both forms have grave restrictions. Obviously, a plane in intercept form is not allowed to pass through the coordinate origin  $\mathbf{0} = (0, 0, 0)$ , and in spherical coordinates there are ambiguities in the angles when a plane passes through this point. Although these forms are able to represent a plane with three parameters, the above drawbacks are not acceptable. Therefore in this thesis a plane is uniquely represented by four parameters: a normal vector  $\mathbf{n}$  perpendicular to its surface, and its smallest distance  $d$  to  $\mathbf{0}$ . The computation of  $\mathbf{n}$  and  $d$  from three vertices of a given triangle is considered as straightforward and therefore not presented here.

### 4.3.2 Sphere

For a sphere the characteristic parameters that need to be determined are

- the centre  $C_{\text{sph}}$  and
- the radius  $r$ .

An estimate for  $r$  is simply obtained from averaging the principal radii of curvature  $R_{\text{max}}$  and  $R_{\text{min}}$  associated with  $T$ , i.e.

$$r = \frac{1}{2} (R_{\text{max}} + R_{\text{min}}). \quad (4.3)$$

An estimate for the centre of the sphere  $C_{\text{sph}}$  can be obtained by shifting the compensated centre  $C^*$  of  $T$  on its surface along the compensated normal  $\mathbf{n}^*$  by the distance  $r$ , i.e.

$$C_{\text{sph}} = C^* + r \mathbf{n}^*, \quad (4.4)$$

where the direction of the shift depends on the sign of the mean radius  $r$ . This is because  $\mathbf{n}^*$  is an estimate for the normal direction at  $C^*$ .

### 4.3.3 Cylinder

The characteristic parameters of a cylinder encompass

- the radius  $r$ ,
- a normalised axis vector  $\mathbf{a}_{\text{cyl}}$ , and
- a reference point  $C_{\text{cyl}}$ .

When the triangles on a triangulated surface are assumed to approximate the surface of a cylindrical segment, then in theory two non-adjacent or three adjacent non-coplanar triangles are sufficient to determine the parameters. This follows from the fact that five points can define a unique instance of a cylinder as is stated in [Roth & Levine 91]. However, as for the sphere the characteristic parameters of a cylinder can be estimated based only upon a single triangle if  $\delta_{\text{max}}$  and  $R_{\text{max}}$  are known.

Examining the principal radii of curvature associated with a triangle on the surface of a cylinder and inverting the larger in magnitude provides an initial estimate for its radius  $r$ :

$$r = R_{\max}. \quad (4.5)$$

Similarly,  $\delta_{\max}$  can be used to gain an initial estimate for  $\mathbf{a}_{\text{cyl}}$ , i.e.

$$\mathbf{a}_{\text{cyl}} = \delta_{\max} / \|\delta_{\max}\|. \quad (4.6)$$

As for the centre of a sphere, a reference point  $\mathbf{C}_{\text{cyl}}$  on the axis of a cylinder can be estimated as the compensated centre  $\mathbf{C}^*$  of the triangle on its surface shifted along the compensated normal  $\mathbf{n}^*$  by the distance  $r$ , i.e.

$$\mathbf{C}_{\text{cyl}} = \mathbf{C}^* + r \mathbf{n}^*, \quad (4.7)$$

where the direction of the shift depends on the sign of curvature. This is because  $\mathbf{n}^*$  is an estimate of the normal at  $\mathbf{C}^*$  for the axis  $\mathbf{a}_{\text{cyl}}$  of the presupposed cylinder resulting in a reasonable reference point  $\mathbf{C}_{\text{cyl}}$ . Of course, after a subsequent optimisation of the characteristic parameters the point  $\mathbf{C}_{\text{cyl}}$  is best to be determined uniquely as the point on  $\mathbf{a}_{\text{cyl}}$  that minimises the distance to  $\mathbf{0}$ .

#### 4.3.4 Cone

Unlike for the previous types of geometric primitives the estimation of the characteristic parameters of a cone, encompassing

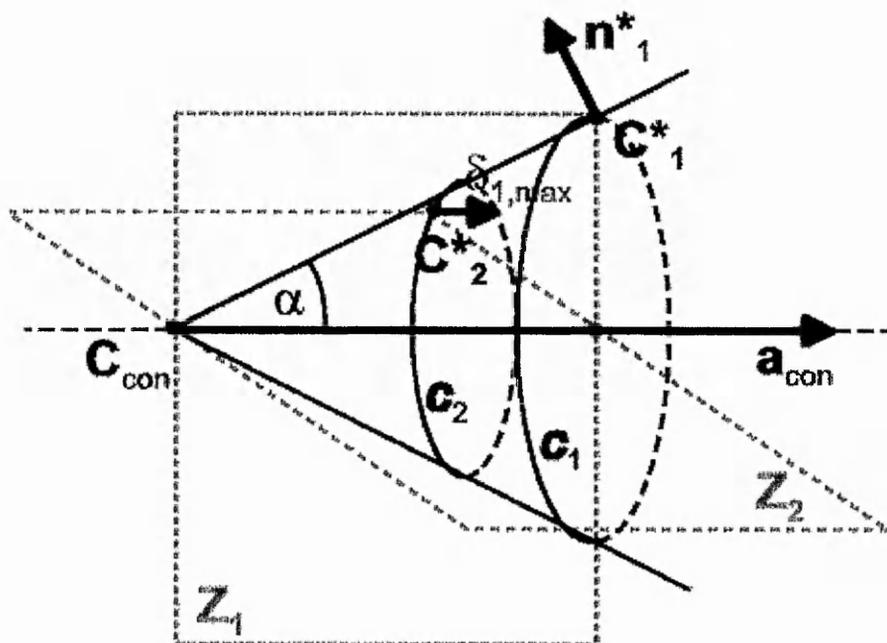
- a unit axis vector  $\mathbf{a}_{\text{con}}$ ,
- the apex  $\mathbf{C}_{\text{con}}$ , and
- the opening angle  $2\alpha$ ,

is a more complicated task. For this, the following data needs to be available: two (not necessarily adjacent) non-coplanar triangles  $T_1$  and  $T_2$  on a triangulated surface with associated compensated centres  $\mathbf{C}^*_i$ , corresponding compensated normals  $\mathbf{n}^*_i$ , principal curvature values  $\kappa_{i, \max}$  and  $\kappa_{i, \min}$ , and, associated with the curvature, principal directions of curvature  $\delta_{i, \max}$  and  $\delta_{i, \min}$ , where  $i = 1, 2$ .

The assumption that the  $T_i$  are located in a convex conical region of a triangulated surface implies  $0 \approx \kappa_{i, \max} > \kappa_{i, \min}$ . However, to simplify the situation for the remainder of this section all curvature values are supposed to have non-negative values only, so that  $\kappa_{i, \max} > \kappa_{i, \min} \approx 0$ . The same inequality holds if the  $T_i$  were located in a concave conical region.

For the following it is important that the difference between  $\kappa_{1, \max}$  and  $\kappa_{2, \max}$  exceeds a small constant value,  $10^{-6}$  say, in magnitude. Otherwise because of numerical reasons the subsequent estimations may not work properly, and thus they need to be repeated with another pair of adjacent triangles. Without loss of generality, another assumption for the corresponding principal radii of curvature  $r_1 = R_{1, \max}$  and  $r_2 = R_{2, \max}$  is:

$$r_2 < r_1.$$



**Figure 4.3.4.1:** *Initial configuration for the determination of characteristic parameters of a cone from two compensated centres  $C^*_i$  of two triangles (not shown), a compensated normal  $\mathbf{n}^*_1$  of the larger triangle, and its direction  $\delta_{1, \max}$  of principal curvature. The plane  $Z_i$  is defined by  $\mathbf{a}_{\text{con}}$ ,  $C_{\text{con}}$ , and  $C^*_i$  ( $i=1, 2$ ). Because in general  $C^*_2$  does not lie in  $Z_1$ ,  $C^*_2$  needs to be projected to  $Z_1$  (or vice versa). Figure 4.3.4.2 shows how to approximate the direction for the projection of  $C^*_2$  onto  $Z_1$ .*

After inspecting the configuration of Figure 4.3.4.1 the idea is to reduce the 3D estimation problem of  $\mathbf{a}_{\text{con}}$ ,  $C_{\text{con}}$ , and  $2\alpha$  to a 2D estimation problem by projection onto a plane. For this, the projection plane  $Z_i$  could be defined by  $\mathbf{a}_{\text{con}}$ ,  $C_{\text{con}}$ , and  $C^*_{3-i}$ , where  $i \in \{1, 2\}$ . In general, the compensated centre  $C^*_2$  would not lie in  $Z_1$  and vice versa. The projection of  $C^*_2$  onto  $Z_1$  would allow an easier estimate of the desired characteristic parameters by further geometric calculations. However, since  $\mathbf{a}_{\text{con}}$  and  $C_{\text{con}}$  are unknown, it is only possible to *approximate the projection*

$C'_2$  of  $C^*_2$  onto  $Z_1$ . Such an approximation can be achieved in two steps as shown in Figure 4.3.4.2. Please note that for a cone that is relatively large compared to the triangles  $T_1$  and  $T_2$  on its surface (represented by its compensated centres) the angle between  $Z_1$  and  $Z_2$  is in general much smaller than illustrated.

So in the first step, the vector  $\mathbf{d}$  in Figure 4.3.4.2 (a) is determined by eliminating components parallel to  $\mathbf{n}^*_1$  to minimise the distance to  $C^*_1$  as

$$\mathbf{d} = \delta_{1, \max} - \lambda \mathbf{n}^*_1, \quad (4.8)$$

where  $\lambda = \langle \delta_{1, \max}, \mathbf{n}^*_1 \rangle$ . Because  $\delta_{1, \max}$  and  $\mathbf{n}^*_1$  are in theory perpendicular to each other,  $\|\mathbf{d}\| > 0$ . This implies that the point  $C'_2$  in 4.3.4.2 (b) lies on the straight line  $C^*_2 + \mu \mathbf{d}$ , such that it minimises the distance to  $C^*_1$ . Moreover, it follows that the vectors  $(C^*_2 + \mu \mathbf{d}) - C^*_2$  and  $C^*_2 + \mu \mathbf{d} - C^*_1$  are perpendicular to each other for some  $\mu \neq 0$ . Thus

$$\begin{aligned} 0 &= \langle (C^*_2 + \mu \mathbf{d}) - C^*_2, C^*_2 + \mu \mathbf{d} - C^*_1 \rangle \\ &= \mu^2 \langle \mathbf{d}, \mathbf{d} \rangle + \mu \langle \mathbf{d}, C^*_2 - C^*_1 \rangle. \end{aligned} \quad (4.9)$$

Hence

$$\mu = \langle \mathbf{d}, C^*_2 - C^*_1 \rangle / \langle \mathbf{d}, \mathbf{d} \rangle, \quad (4.10)$$

and consequently

$$C'_2 = C^*_2 + (\langle \mathbf{d}, C^*_2 - C^*_1 \rangle / \langle \mathbf{d}, \mathbf{d} \rangle) \mathbf{d}. \quad (4.11)$$

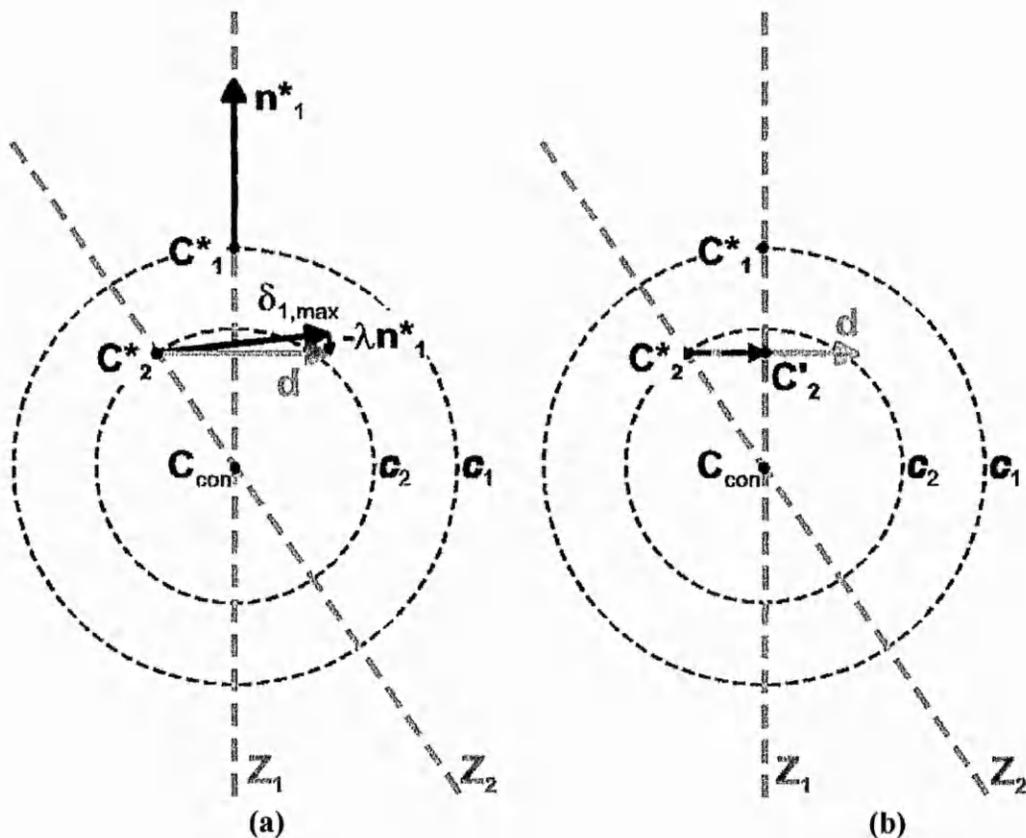
The following estimations can be established readily with some help of Figure 4.3.4.3:

$$C_{\text{con}} = C^*_1 + (C'_2 - C^*_1) \left| r_1 / (r_1 - r_2) \right|, \quad (4.12)$$

$$\alpha = \cos^{-1}(\|C^*_1 - C'_2\| / \|D - E\|), \quad (4.13)$$

$$\mathbf{a}_{\text{con}} = (C_{\text{con}} - E) / \|C_{\text{con}} - E\|, \quad (4.14)$$

where  $\mathbf{B} = C^*_1 + r_2 \mathbf{n}^*_1$ ,  $\mathbf{D} = C'_2 + r_2 \mathbf{n}^*_1$ , and  $\mathbf{E} = C^*_1 + r_1 \mathbf{n}^*_1$ . These estimations make sense because  $\mathbf{E}$  is the centre of the osculating sphere through  $C^*_1$  with radius  $r_1$ , and therefore  $\mathbf{E}$  is located on the cone's axis (this is also used in [Lukács et al. 98]). This method for estimating the characteristic parameters of a cone is later on referred to as method A.



**Figure 4.3.4.2:** *Determination of an approximate projection of  $C^*_2$  to the point  $C'_2$  in plane  $Z_1$  of Figure 4.3.4.1 using the two given compensated centres  $C^*_i$ , a compensated normal  $n^*_1$ , and a direction of minimum principal curvature  $\delta_{1,max}$  (projection along axis of the cone). The circles  $c_1$  and  $c_2$  correspond to the ones in Figure 4.3.4.1.*

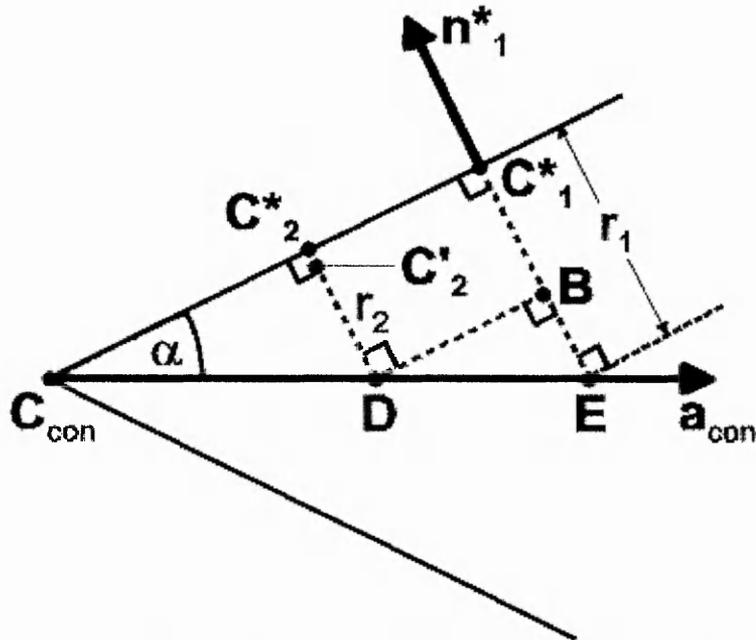
- (a) *Compute the direction  $d$  using the component of  $\delta_{1,max}$  perpendicular to  $n^*_1$*
- (b) *Determine  $C'_2$  as the point on the line  $C^*_2 + \mu d$  that minimises the distance to  $C^*_1$*

Alternatively, the opening angle  $2\alpha$  can be estimated by means of the equation

$$\alpha = \cos^{-1}(\|C^*_1 - C_{con}\| / \|C_{con} - E\|). \quad (4.15)$$

Retaining the estimations of the other characteristic parameters this method is denoted as method B for estimating the parameters of a cone.

An example in Section 4.6.3 shows that method B produces less accurate results than method A used for the example in Section 4.6.2. A brief discussion of the approximation is given in Section 4.7.2.



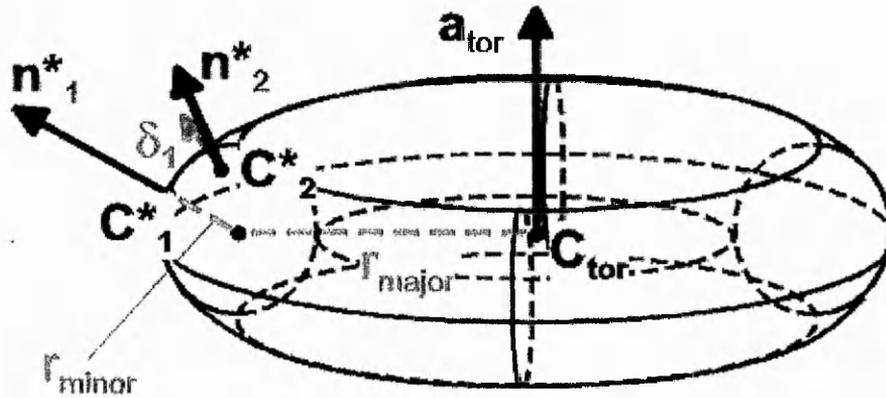
**Figure 4.3.4.3:** *Estimating the characteristic parameters of a cone exploiting the similarity between the triangles  $(C^*_1, C_{con}, E)$ ,  $(C^*_2, C_{con}, D)$ , and  $(B, D, E)$  in the plane  $Z_1$ , where  $C^*_2$  is approximated by  $C'_2$ .*

### 4.3.5 Torus

As for the cone estimates of the characteristic parameters of a torus can be determined by elementary geometric calculations. These parameters are:

- a unit axis vector  $\mathbf{a}_{tor}$  (the “main axis”) perpendicular to the central symmetry plane,
- the centroid  $C_{tor}$  (centre of gravity),
- the major radius  $r_{major}$  (“inner radius”, see Figure 4.3.5.1), and
- the minor radius  $r_{minor}$  (radius of the ring, see Figure 4.3.5.1).

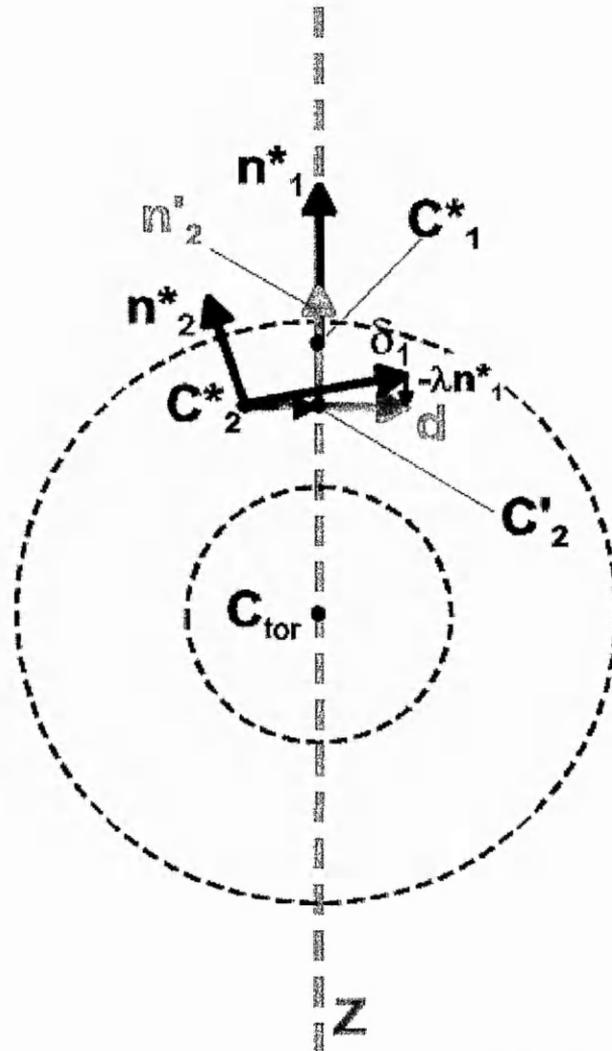
These parameters may be estimated by employing the compensated centres  $C^*_i$ , the compensated normals  $\mathbf{n}^*_i$ , the principal radii of curvature  $R_{i, max}$  and  $R_{i, min}$  ( $i = 1, 2$ ) as well as the principal directions  $\delta_{i, min}$  and  $\delta_{i, max}$  of two adjacent, non-coplanar triangles  $T_i$  on a triangulated surface (as in Figure 4.3.5.1).



**Figure 4.3.5.1:** *Characteristic parameters of a torus (central axis  $a_{tor}$ , centroid  $C_{tor}$ , major radius  $r_{major}$ , and the minor radius  $r_{minor}$ ) with compensated centres  $C^*_i$  of two adjacent triangles (not shown), corresponding normals  $n^*_i$ , and a direction of principal curvature  $\delta_i$  for the second triangle. Other principal radii of curvature, curvature directions and the triangles  $T_i$  are not shown.*

The first problem to overcome (referred to as “correspondence problem” below) is to determine for each of the  $T_i$  which of its associated principal radii correspond to the major and which to the minor radius of the torus. This problem can potentially be solved by the following observation: though the distances from the  $C^*_i$  to the main axis can vary, the distances of the  $C^*_i$  to the circular centre line in the “bulge” of the torus are expected to have a nearly constant value, namely  $r_{minor}$ . Thus, either  $R_{1,max}$  or  $R_{1,min}$  of  $T_1$  corresponds to  $r_{minor}$ , and the same applies to either  $R_{2,max}$  or  $R_{2,min}$ . By picking the pair of the principal radii among all possible pairings (one radius from each triangle) that minimises the difference between them, it is likely to obtain a suitable definition of  $r_{minor}$  as the average of both. This has been the method of choice in this thesis. In order to simplify the notation, the principal radii of  $T_1$  and  $T_2$  corresponding to  $r_{minor}$  are denoted by  $r_1$  and  $r_3$ , respectively, so that  $r_{minor} \approx r_1 \approx r_3$ , and their counterparts with  $r_2$  and  $r_0$ , respectively (see Figure 4.3.5.3).

If the above minimum of the differences between the principal radii emerges not to be “clearly unique”(i.e. if more than just one pair of the principal radii yields about the same difference), the  $T_i$  need to be discarded as a seed region for the algorithm presented in Figure 4.2.1, and another pair of triangles is selected instead.



**Figure 4.3.5.2:** *Determination of an approximate projection of  $C^*_2$  to the point  $C'_2$  and a compensated normal  $\mathbf{n}^*_2$  to the normal  $\mathbf{n}'_2$  in plane  $Z$  on a torus from two given compensated centres  $C^*_i$ , a compensated normal  $\mathbf{n}^*_i$ , and a direction of principal curvature  $\delta_i$  (the axis is perpendicular to the page). The required calculations for  $C'_2$  is identical to those for  $C'_2$  as for the cone in Figure 4.3.4.2, and similarly  $\mathbf{n}'_2$  is calculated.*

Now the solution to the correspondence problem allows to identify a principal curvature direction  $\delta_1$  that is almost perpendicular to the torus' axis  $\mathbf{a}_{\text{tor}}$  (such as in Figure 4.3.5.2). Moreover, the problem to estimate characteristic parameters in 3D can be reduced to a 2D problem, as for the parameters of the cone (see again Figure 4.3.4.3). For this, the determination of an approximate projection plane  $Z$  is required that passes through the compensated normal  $\mathbf{n}^*_1$  of  $T_1$  and  $\mathbf{a}_{\text{tor}}$ . With this notion projections of  $C^*_2$  and  $\mathbf{n}^*_2$  into  $Z$  as  $C'_2$  and  $\mathbf{n}'_2$ , respectively, can be used for further geometric calculations.

Next, let  $\delta_2$  be the direction that is associated with  $r_2$ . Then the vector

$$\mathbf{d} = \delta_1 - \lambda \mathbf{n}^*_1, \quad (4.16)$$

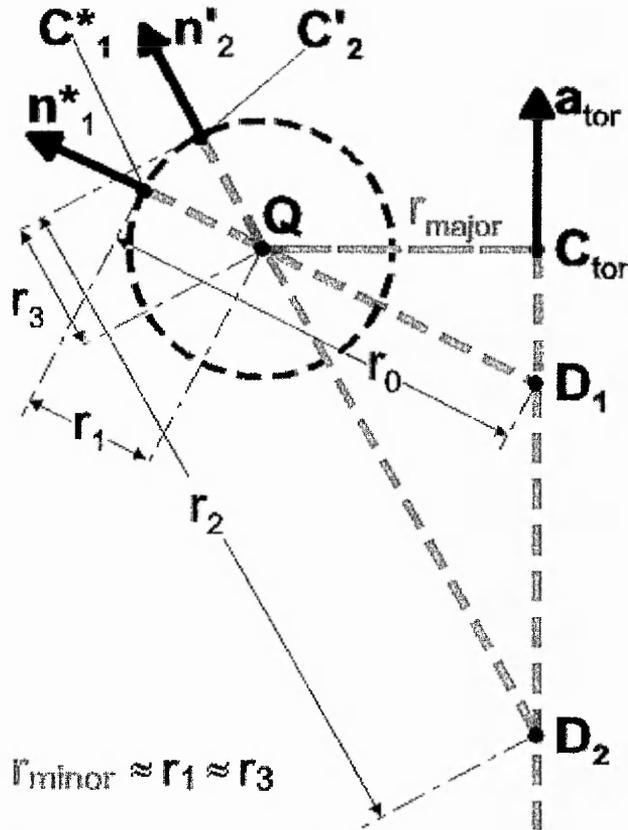
in Figure 4.3.5.2 can be computed as in the previous subsection, and  $\lambda = \langle \delta_1, \mathbf{n}^*_1 \rangle$ . If  $\mathbf{d}$  is the null vector, then set  $\mathbf{C}'_2 = \mathbf{C}^*_2$  and  $\mathbf{n}'_2 = \mathbf{n}^*_2$ . Otherwise obtain the projection of  $\mathbf{C}^*_2$  in the same fashion as in the previous section from

$$\mathbf{C}'_2 = \mathbf{C}^*_2 + \mu \mathbf{d}, \quad (4.17)$$

and similarly

$$\mathbf{n}'_2 = (\mathbf{n}^*_2 + \mu \mathbf{d}) / \|\mathbf{n}^*_2 + \mu \mathbf{d}\| \quad (4.18)$$

can be calculated as a normalised vector, where  $\mu = \langle \mathbf{d}, \mathbf{C}^*_2 - \mathbf{C}^*_1 \rangle / \langle \mathbf{d}, \mathbf{d} \rangle$ .



**Figure 4.3.5.3:** *Determination of the characteristic parameters of the torus employing  $\mathbf{C}^*_1$ ,  $\mathbf{C}'_2$ ,  $\mathbf{n}^*_1$ ,  $\mathbf{n}'_2$ , and the radii  $r_j$  ( $j = 0, \dots, 3$ ) corresponding to the principal radii of curvature (cross-sectional view shows only a single side of the ring indicated by the dashed circle; see text for details).*

For the next estimates it is assumed that all of the principal radii are finite. If one of these radii is infinite (i.e. one of the triangle normals is parallel to the main axis of the torus), the argumentation is basically the same, but some formulae require a

modification (see further below). If two principal radii are infinite, the present pair of triangles  $T_i$  is discarded and another pair of seed triangles needs to be selected.

So assuming the first case, estimates for the support points  $\mathbf{D}_1$ ,  $\mathbf{D}_2$  and  $\mathbf{Q}$  in Figure 4.3.5.3 are obtained from

$$\mathbf{D}_1 = \mathbf{C}^*_1 + r_0 \mathbf{n}^*_1, \quad (4.19)$$

$$\mathbf{D}_2 = \mathbf{C}'_2 + r_2 \mathbf{n}'_2, \quad (4.20)$$

$$\mathbf{Q} = \frac{1}{2} (\mathbf{C}^*_1 + r_1 \mathbf{n}^*_1 + \mathbf{C}'_2 + r_3 \mathbf{n}'_2). \quad (4.21)$$

The point  $\mathbf{Q}$  is supposed to be on the circular centre line in the “bulge” of the torus, whereas  $\mathbf{D}_1$  and  $\mathbf{D}_2$  are expected to lie approximately on the main axis  $\mathbf{a}_{\text{tor}}$  passing through the centroid  $\mathbf{C}_{\text{tor}}$ . As for the cone, the above estimations make sense because  $\mathbf{D}_1$  and  $\mathbf{D}_2$  are the centres of osculating spheres through  $\mathbf{C}^*_1$  (with radius  $r_0$ ) and  $\mathbf{C}'_2$  (with radius  $r_2$ ), respectively.

With these preparations the characteristic parameters of the torus can be estimated as follows:

$$\mathbf{a}_{\text{tor}} = (\mathbf{D}_2 - \mathbf{D}_1) / \|\mathbf{D}_2 - \mathbf{D}_1\|, \quad (4.22)$$

$$\mathbf{C}_{\text{tor}} = \mathbf{D}_1 + \langle \mathbf{Q} - \mathbf{D}_1, \mathbf{a}_{\text{tor}} \rangle \mathbf{a}_{\text{tor}} \quad (4.23)$$

as the point on  $\mathbf{a}_{\text{tor}}$  when it passes through  $\mathbf{D}_1$  and minimises the distance to  $\mathbf{Q}$  (see Figure 4.3.5.3),

$$r_{\text{major}} = \|\mathbf{C}_{\text{tor}} - \mathbf{Q}\|, \quad (4.24)$$

and

$$r_{\text{minor}} = \frac{1}{2} (r_1 + r_3). \quad (4.25)$$

As can be deduced from Figure 4.3.5.3 the estimates become bad when the radii  $r_0$  and  $r_2$  are large in magnitude. So when the corresponding principal curvature values are small in magnitude, the estimation is best to be repeated with another pair of adjacent, non-coplanar triangles that have larger principal curvatures in magnitude.

Now the special case is discussed where one triangle normal is parallel to the main axis  $\mathbf{a}_{\text{tor}}$ , meaning that one of the corresponding radii (either  $r_0$  or  $r_2$ ) is at least nearly infinite. Without loss of generality, it is assumed that this normal is  $\mathbf{n}^*_1$ ; otherwise the roles of  $T_1$  and  $T_2$  are swapped. Accordingly,  $\mathbf{n}^*_1$  can be taken as an estimate for the main axis, i.e.

$$\mathbf{a}_{\text{tor}} = \mathbf{n}^*_1. \quad (4.26)$$

Furthermore, similar to the configuration as in Figure 4.3.5.3 this implies

$$\mathbf{D}_2 = \mathbf{C}'_2 + r_2 \mathbf{n}'_2 \quad (4.27)$$

and

$$\mathbf{Q} = \mathbf{C}'_2 + r_3 (\mathbf{D}_2 - \mathbf{C}'_2) / \|\mathbf{D}_2 - \mathbf{C}'_2\|, \quad (4.28)$$

so an estimate of  $\mathbf{C}_{\text{tor}}$  can be obtained from

$$\mathbf{C}_{\text{tor}} = \mathbf{D}_2 + \langle \mathbf{Q} - \mathbf{D}_2, \mathbf{a}_{\text{tor}} \rangle \mathbf{a}_{\text{tor}}. \quad (4.29)$$

As before, the major radius can be estimated by

$$r_{\text{major}} = \|\mathbf{C}_{\text{tor}} - \mathbf{Q}\|, \quad (4.30)$$

and the minor radius

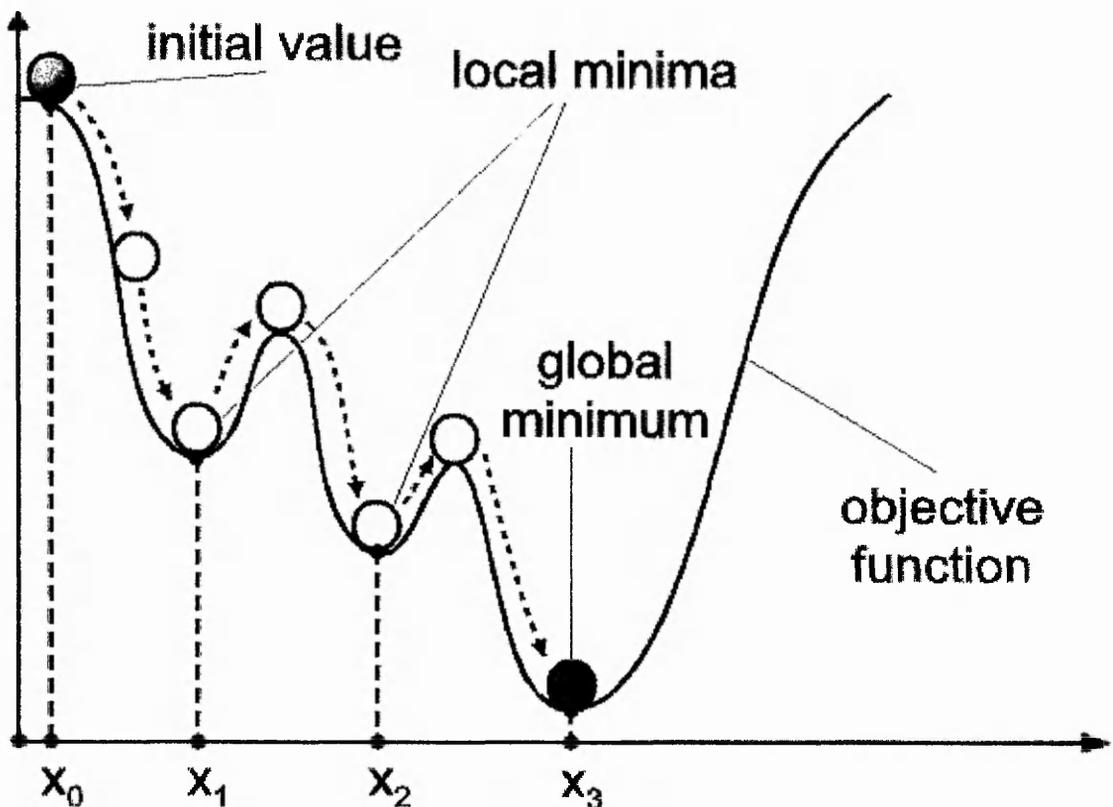
$$r_{\text{minor}} = r_3. \quad (4.31)$$

#### 4.4 Methods of parameter optimisation

The next crucial step after the initial determination of characteristic parameters for each of the geometric primitives concerns their adjustment after a growth step. As can be seen in the modified region growing algorithm in Section 4.2, it is attempted to determine a vector  $\mathbf{X}_g$  of characteristic parameters of a geometric primitive that fits to the data points of the current region within a small tolerance. The aim is to optimise the goodness of fit that is formulated in this thesis by Equation (2.8) as a minimax problem. Thus the determination of  $\mathbf{X}_g$  involves numerical optimisation, where an objective function that reflects the goodness of fit needs to be minimised.

A problem arises from the existence of usually many local minima of the objective function (see Figure 4.4.1) that causes some optimisation methods to

converge to an unwanted non-optimal solution. The problem may be even more exacerbated if the objective function is highly non-linear or even not differentiable. However, methods not involving derivatives that solve nonlinear and non-smooth minimisation problems (such as Direct Search methods, GAs, and tabu search) have briefly been introduced in Section 2.4. In the following two representants of these methods, a GA and a Direct Search optimisation, are presented in more detail. Moreover, it is investigated how these methods can be employed to solve the surface fitting problem.



**Figure 4.4.1:** *Local minima may cause an optimisation method to converge to an unwanted solution (such as  $x_1$  or  $x_2$ ). The convergence to a local minimum depends on the method, its initial value and the objective function.*

#### 4.4.1 Parameter optimisation employing a Genetic Algorithm

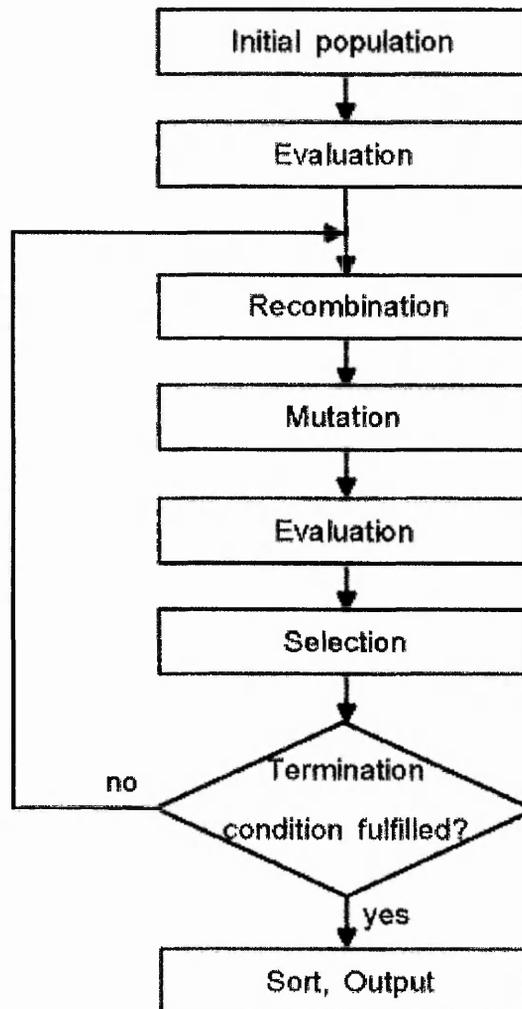
The objective function used in this thesis (Equation (4.32) in Section 4.4.1.2) involves a non-differentiable maximum function, so that in general partial derivatives do not exist. For this reason many of the common optimisation methods that are based on availability of derivatives are not applicable. As explained in Section 2.4.2, a Genetic Algorithm (GA) does not require the

derivatives of the objective function. Moreover, it has the capability to escape from local minima (which typically exist in abundance) with a greater chance to detect global minima. For this, such an algorithm simulates the evolutionary principle of the optimisation of species from generation to generation by genetic reproduction. In the physical world the genes of the “fittest” individuals survive with a higher probability because their information enable their host to be better adapted to a specific environment. When this genetic information is bequeathed from parents to a child, mechanisms of genetic recombination and mutation provide a genetic variation. Thus the genes of the child differ slightly from those of its parents. A child with “good genes” is likely to have increased individual fitness and hence higher chance of further reproduction.

The biological principle of genetic optimisation has been transferred into algorithms by many researchers (of which [Holland 75] is one of the earliest) that solve optimisation problems in automated data processing. So the principles of optimising with GA's are explained next, and then the requirements of the current project are specified.

#### **4.4.1.1 Principles of Genetic Algorithms**

The aim of the optimisation with a GA is in general the determination of an optimal (maximal or minimal) solution of a real valued objective function over a multidimensional search space  $\Omega$ . Each point in  $\Omega$  stands for a biological individual, where each coordinate component of a point represents a gene of the individual. So the vector of all coordinate components of a point forms a sequence of genes that defines a *chromosome* of the individual. The set of all individuals at a time  $t$  is denoted as population. The number of individuals `POPULATION_SIZE` in the population is an algorithmic parameter, and its setting for this project is explained in Section 4.5. In a black box scheme the optimisation using GA's can be structured as is illustrated in Figure 4.4.2 (adapted from [Fischer 99]).



**Figure 4.4.2:** *Optimisation using Genetic Algorithms in a black box scheme (adapted from [Fischer 99])*

At the time  $t = 0$ , a limited *initial population* is generated, where the genes of the individuals may be initialised randomly or by exploiting prior knowledge about the optimisation problem. The fitness of the individuals of the initial population is then determined by an *evaluation* of the objective function. Afterwards a loop begins that is terminated if a certain condition is fulfilled (for example, when a preset number of generations is reached). Within the loop the genes of the individuals of the population at the time  $t$  are subject to a *recombination* in order to create a new population at the time  $t+1$ . Here only the genes of individuals showing a sufficiently good fitness may be selected for the recombination. Individuals of population  $t+1$  obtain the genes of their parents by a crossing of the parental genes. Hereafter the crossed genes are subject to a *mutation* process, where individual genes are randomly selected and modified. Then follows an *evaluation* of the fitness of the individuals of the new population at the time  $t+1$ ,

and a *selection* of the best ones is employed to form the next population. Individuals not selected are discarded. The loop is then repeated for the new population, until the *termination condition* is fulfilled. The individuals of the current population are *sorted* depending on their fitness and *output*.

For the realisation of a GA the following design steps need to be considered:

1. Selection of an appropriate objective or “fitness” function
2. Determination of a termination criterion
3. Encoding of genes and chromosomes
4. Specification of the genetic operators recombination, selection, mutation
5. Generation of an initial population

The next section explains how these steps are realised to solve the surface fitting problem of this project.

#### 4.4.1.2 Surface fitting with a Genetic Algorithm

The **objective function** that is to be minimised reflects the maximum Euclidean distance of all the points to the current surface region that is in turn represented by a vector  $\mathbf{X}$  of characteristic parameters for a geometric primitive  $g \in \mathbf{G}$ . So by repeating Equation (2.8) the objective function is

$$d(\mathbf{R}, \mathbf{X}) = \max_{\mathbf{P} \in \mathbf{R}} |f(\mathbf{P}; \mathbf{X})|, \quad (4.32)$$

where  $\mathbf{R}$  denotes the region currently grown in  $\mathbf{S}$ , and  $f$  models the distance of  $\mathbf{P}$  to the surface of  $g$ .

Though other **termination conditions** are possible, this work terminates the optimisation process if a predefined number of generations is reached. This number is denoted as GENERATIONS and belongs to the algorithmic parameters that are described in detail in Section 4.5; this section also includes values for all algorithmic parameters used in this thesis. Possible alternative conditions involve the termination when exceeding a given computational time (particularly for time-critical applications) and the stagnation of the iterated parameters for a preset number of generations.

The **encoding of chromosomes** is provided by an  $n$ -dimensional vector in the parameter space, where  $n$  is the number of real values required to represent a unique instance of a geometric primitive. A sphere, for example, is represented by three coordinate values for the centre and one value for the radius as is described in Section 4.3.2. This defines a point in a 4-dimensional space, i.e. a sequence of genes of four components. Similarly, the characteristic parameters of unique instances of cylinder, cone, and torus describe specific points in a 7-, 7-, and 8-dimensional space, respectively.

The next three paragraphs explain the strategies for recombination, mutation, and selection that are employed for this work.

Usually two ways of the **recombination** of two parental chromosomes are used, where each child gene is either obtained by averaging each pair of corresponding parental genes, or a random selection of one of the parental genes. This project applies both methods as follows. Depending on the value of a random variable  $rnd$  that is uniformly distributed in the interval  $[0, 1)$ , a child gene is obtained by

averaging	if $rnd \geq 0.5$ , and
random selection	otherwise.

Again depending on  $rnd$ , the random selection selects as a child gene the parental gene from the

first individual	if $0 \leq rnd < 0.25$ , and from the
second individual	if $0.25 \leq rnd < 0.5$ .

In order to escape from local optima, **mutation** plays a key role in a GA. For the initial generations it is desirable to obtain a variety of genes in order to escape from local minima. On the other hand, near a global optimum the mutation rate should be lower. Hence for this work a mutation is performed on the entire chromosome of an individual for about the first 10% of GENERATIONS, and later merely a randomly chosen single gene of a chromosome is mutated. The selection of an individual in a population is realised by a loop over all individuals of the generation  $t$ . Depending on the value of a random variable  $rnd2$  uniformly distributed in the unit interval  $[0, 1)$ , the chromosome of the individual is subject

to mutation, if  $rnd2$  is smaller than a preset algorithmic parameter  $MUTATION\_RATE$ . Each gene  $g$  of the chromosome that is selected for mutation is manipulated as follows. It is assumed that  $b$  is a positive value initialised by the algorithmic parameter  $UPPERBOUND$ , and that  $N(0, 1)$  is a normally distributed random number with expected value 0 and standard deviation 1. Then  $g$  is replaced by

$$g' = g + b * N(0, 1), \quad (4.33)$$

and  $b$  is updated by  $b' = b * MUTATION\_FACTOR$ , where the latter algorithmic parameter is a positive number slightly smaller than 1.

After a new generation  $t+1$  has been generated, the best individuals are subject to a **selection** step. There are two alternatives: either the best individuals are selected from generation  $t$  *and* generation  $t+1$  (i.e. the parents have a chance to survive), or the selection merely considers the individuals of generation  $t+1$ . For this project the first alternative has been chosen. The selection itself is strongly based on the fitness of the individuals. Though some selection methods give individuals with a small fitness value still a chance, in this thesis a strict elitism is applied, such that only the fittest individuals survive. Their number amounts to  $BEST\_NUM$ .

Each GA requires the generation of an **initial population**. The number of its individuals is given by the algorithmic parameter  $POPULATION\_SIZE$ . The genes of its individuals may be initialised with random numbers or preset values. For this project every chromosome of the first population is initialised with the characteristic parameters that have been estimated in Section 4.3 for each type of geometric primitive except planes. Planes are extracted by a purely geometric method explained in Chapter 5. Then each chromosome (except one) is subject to a mutation similar as above where each gene  $g$  in a chromosome is replaced by

$$g' = g + 0.01 \cdot MUTATION\_VARIANCE \cdot N(0, 1). \quad (4.34)$$

#### 4.4.2 Parameter optimisation employing Direct Search

As pointed out before, direct minimisation methods do not require information about derivatives of  $f$ . Amongst such methods is the one of [Hooke & Jeeves 61] that has been absorbed by [Bronstein & Semendjajew 85] and is quoted here for future reference:

“This method uses as a basic operation the »exploratory search from a point  $\mathbf{x} = (x_1, \dots, x_n)^T$  with a step width  $\lambda$  and the comparison value  $F$ «:

The condition  $f(x_1 + \lambda, x_2, \dots, x_n) < F$  is tested,  
if YES  $[\mathbf{x}]^1 = (x_1 + \lambda, x_2, \dots, x_n)^T$ ,  
otherwise the condition  $f(x_1 - \lambda, x_2, \dots, x_n) < F$  is tested,  
if YES  $[\mathbf{x}]^1 = (x_1 - \lambda, x_2, \dots, x_n)^T$ ,  
else  $[\mathbf{x}]^1 = \mathbf{x}$ .

Then from  $[\mathbf{x}]^1$  onward by  $\lambda$ -steps the  $x_2$ -direction is to be searched exploratory, result  $[\mathbf{x}]^2$ , etc, until  $[\mathbf{x}]^n$ .

**Algorithm:** Given a starting point  $\mathbf{x}$ , a step width  $\lambda$  and an  $\alpha$  such that  $0 < \alpha < 1$ .

- I. Search exploratory from  $\mathbf{x}$  onward with step width  $\lambda$  and the comparison value  $F = f(\mathbf{x})$ . Is  $\mathbf{x} \neq [\mathbf{x}]^n$ ?  
YES: go to II;  
NO: take  $\alpha\lambda$  as the new  $\lambda$  and go to I.
- II. Search exploratory from  $[\mathbf{x}]^n + ([\mathbf{x}]^n - \mathbf{x})$  onward with step width  $\lambda$  and the comparison value  $F = f([\mathbf{x}]^n)$ . Is  $[\mathbf{x}]^n + ([\mathbf{x}]^n - \mathbf{x}) \neq [[\mathbf{x}]^n + ([\mathbf{x}]^n - \mathbf{x})]^n$  satisfied?  
YES: take  $[\mathbf{x}]^n$  as the new  $\mathbf{x}$ ,  $[\mathbf{x}]^n + ([\mathbf{x}]^n - \mathbf{x})$  as the new  $[\mathbf{x}]^n$ , then go to II;  
NO: take  $[\mathbf{x}]^n$  as the new  $\mathbf{x}$ , take  $\alpha\lambda$  as the new  $\lambda$ , go to I.

The computation will be finished as soon as in I. the case NO occurs and  $\lambda$  is already sufficiently small.

*Remark: This method and its advancements lead quickly to the vicinity of the minimum. ...*”

So in a nutshell, the minimisation method of Hooke and Jeeves attempts to find a minimum in a vicinity of a point  $\mathbf{x}$ , where this vicinity consists of the sum of  $\mathbf{x}$  and a set of directions  $\{\mathbf{d}\}$ . Each vector  $\mathbf{d}$  in turn is the sum of the straight lines that are parallel to the directions of the coordinate axes with a length  $\lambda > 0$ , such that

$$f(\mathbf{x} + \lambda \mathbf{d}) < f(\mathbf{x}). \quad (4.35)$$

Once such a point  $\mathbf{x} + \lambda \mathbf{d}$  in the vicinity of  $\mathbf{x}$  has been determined, it becomes the new  $\mathbf{x}$ , and the length  $\lambda$  is decreased by multiplication with a factor in the interval  $[0, 1)$ . The algorithm terminates, if no smaller value than  $f(\mathbf{x})$  can be found in a vicinity, and  $\lambda$  is below a very small positive threshold,  $10^{-6}$  say.

Obviously, after another look at Figure 4.4.1, this minimisation method is not designed to escape from local minima.

### **4.4.3 Parameter optimisation by a combined method**

[Fischer 99] employed a GA for the determination of a translation vector and a rotation matrix describing the transformation of a polyhedral surface that has been measured from two distinct views. One of his results (translated by the Author) says that *“it can be observed that the method converges already after a few generations to the global optimum, and then needs essentially more generations for the fine optimisation.”*

This result suggests to combine an optimisation method that is able to escape from a local optimum and detects a global one with such a method that converges quicker. In this project it is necessary that both methods do not require functional derivatives. Consequently, in this thesis a GA has been applied to determine a point  $\mathbf{x}^*$  in the “optimal valley of the objective function” (such as the valley of the point  $\mathbf{x}_3$  in Figure 4.4.1), and then  $\mathbf{x}^*$  is employed as a starting point for the Direct Search method presented in Section 4.4.2.

The Direct Search needs further two algorithmic parameters, of which one is  $\alpha \in ]0, 1)$  and the other is the initial step length  $\lambda > 0$ . These parameters are explained in the next section, and are denoted as STEP\_FACTOR and STEP\_WIDTH, respectively.

## **4.5 Algorithmic parameters and their settings**

This section explains the most relevant algorithmic parameters controlling the segmentation process. These algorithmic parameters are grouped into three

categories: general parameters, parameters specific for the optimisation employing a GA, and parameters specific for a Direct Search optimisation method. The settings of these parameters are listed and remain valid for all processed segmentation results unless explicitly specified otherwise.

#### **4.5.1 General algorithmic parameters**

DIAMETER is a value of a triangulated surface denoting the maximum distance between all possible pairs of surface vertices. Although this value depends on the shape of the surface, it is - in contrast to a bounding box of a triangulated surface - invariant to translations and rotations. Hence it is used as a factor to remove scale-dependencies of other algorithmic parameters.

CURVATURE is a parameter classifying triangles in two categories of estimated mean curvature on a triangulated surface. A triangle of estimated mean curvature not exceeding CURVATURE obtains the tag LOW, otherwise the tag HIGH in step 1 of the modified region growing algorithm in Figure 4.2.1. Empirically the value of CURVATURE is set to  $2.0 / \text{DIAMETER}$ .

$\tau$  is a tolerance parameter controlling the quality of a segment of a geometric primitive which is extracted from a triangulated surface. Each data point belonging to the surface of the segment needs to be within a Euclidean distance  $\tau$  of the surface (see Figure 4.2.1, step 2.A.d.iv). Although this parameter might be initialised by a value of about  $0.001 \cdot \text{DIAMETER}$ , say, such a preset value would in general not reflect the amount of noise in the surface data. A more elegant method to determine the value of this parameter has been suggested by [Besl & Jain 88]: fit a plane locally to the data and take the least-squares point-to-plane distance as a value for  $\tau$ . Based on this idea, a similar method for the automated determination of  $\tau$  is presented in Chapter 5.

MIN\_TRIANGLES controls the minimum number of triangles a primitive segment must have to be accepted as valid, and is used in Figure 4.2.1, step 2.A.e. This parameter aims to prevent a triangulated surface from being split into many small segments (as is used, for example, in [Powell et al. 98]). The problem with an automatic determination of this parameter results from the fact that triangulated

surfaces may vary significantly in the number of surface triangles *and* in their shape characteristics. For this reason, the MIN\_TRIANGLES has been set for each surface individually according to the rating of the user. Values are given with the examples of the segmentation results.

#### **4.5.2 Algorithmic parameters specific for a Genetic Algorithm**

GENERATIONS determines the number of generations of the GA in Section 4.4.1.2 that adjusts a vector of characteristic parameters in order to fit to a given region (represented by a set of data points) during region growing (Figure 4.2.1, step 2.A.d.iv.I). The larger the number is, the higher is the probability that the objective function outputs parameters for which the function has managed to escape from local minima. Because the number of local minima of the objective function can be very high the number of generations for the segmentation results in this thesis has been set to 300 (unless stated otherwise).

MUTATION\_RATE controls the number of individuals of a population whose chromosomes are subject to mutation. This work uses a value of 0.3 as suggested by the work of [Fischer 99] corresponding to about 30% of individuals that are affected by mutation.

MUTATION\_VARIANCE measures the variance of the mutation of a single gene in a chromosome. The genetic mutation itself is performed by adding a normally distributed random number with expected value 0 multiplied by a mutation factor  $b$  that is initialised with MUTATION\_VARIANCE (see Section 4.4.1.2). A value for this parameter has been empirically determined as  $0.75 \cdot \text{DIAMETER}$ .

MUTATION\_FACTOR also characterises the extent to which the mutation of a single gene is allowed to happen. This factor is used in connection with MUTATION\_VARIANCE and the above mutation factor  $b$  to reduce the mutation interval iteratively. Raising this value (e.g. 0.997) to the power of generations (the number of GENERATIONS, e.g. 1000) reduces the value of  $b$  to about  $0.05 \cdot b$  which restricts the mutation of a gene near the global optimum. However, experiments have shown that in most cases the above number of 300 generations has been sufficient to achieve approximately the same result, so that

MUTATION\_FACTOR can be set to 0.99 to achieve approximately the same reduction of  $b$  to  $0.05 \cdot b$ . Values of this parameter are produced with the segmentation results.

POPULATION\_SIZE specifies the number of individuals per generation. The larger the number is, the wider is the variety of chromosomes. However, many applications do not need a huge number since otherwise the computational time increases dramatically. Hence this number is set to 50 in this thesis.

BEST\_NUM specifies the number of individuals that are selected for recombination, and is in this work also used to determine the number of individuals that survive from generation  $t$  to generation  $t+1$ . Empirically, it is set to the largest integer number  $\leq 0.25 \cdot \text{POPULATION\_SIZE}$ .

### **4.5.3 Algorithmic parameters specific for the Direct Search method**

STEP\_WIDTH determines the width of an exploratory step in the Direct Search method in Section 4.4.2, where the parameter is denoted by  $\lambda$ . In this work, a value of  $0.75 \cdot \text{DIAMETER}$  has been empirically determined. Please note that the role of STEP\_WIDTH resembles the one of MUTATION\_VARIANCE for the optimisation with the GA.

STEP\_FACTOR constantly reduces the width of an exploratory step by its value each time after the components of a search direction vector have been determined along all possible axes of the search space. In Section 4.4.2 the parameter is denoted by  $\alpha$ , and in this work a value of 0.86 has been empirically determined. Please note that the role of STEP\_FACTOR resembles the one of MUTATION\_FACTOR for the optimisation with the GA.

## **4.6 Examples of segmentation results**

A segmenter has been developed based on the methods and algorithms described in Sections 4.2 - 4.5 and it has been applied to a selection of triangulated surfaces. This section presents segmentation results as an output for each of these

surfaces including figures of extracted geometric primitives, and a table with their corresponding characteristic parameter values. Each of the numbers has been rounded to five digits precision after the decimal point.

Next, it is explained in further detail how to interpret the segmentation results shown in the figures throughout the remainder of Section 4.6.

- The order of extracting the geometric primitives (unless stated otherwise) is plane, sphere, cylinder, cone, and torus.
- Planes have been extracted employing the geometric method explained in Chapter 5.
- A colour scheme has been used to encode the types of geometric primitives, where each triangle of a segment has a brighter upper side (owing to reflection of light from a virtual light source) and a darker reverse side. Moreover, each segment is enclosed by a black boundary curve. The colours of the types of geometric primitives are:
  - plane: orange (upper) and red (reverse);
  - sphere: green (upper) and dark green(reverse);
  - cylinder: blue (upper) and dark blue(reverse);
  - cone: yellow (upper and reverse);
  - torus: purple (upper and reverse).
- Planes obtain the label PL, spheres SP, cylinders CY, cones CO, and tori TO. The label also consists of a number referring to a particular instance of the geometric primitive.
- Very small segments occurring in all figures and the unique segments of Example 4.6.1 have not been labelled.

It is recalled that all computations and the visualisation have been accomplished by the POMOS software tool.

#### 4.6.1 Segmentation of a hemisphere attached to a plane

As a first simple example the triangulated surface of a hemisphere attached to a plane in Figure 4.6.1 (a) has been measured with a tactile scanner; further details are unknown. A tolerance value of 0.034483 has been automatically determined using the adaptive tolerance method described in Section 5.2.2 for the extraction of a planar and a spherical segment. Further algorithmic parameters are set to:

- MIN\_TRIANGLES = 20,
- GENERATIONS = 300, and
- MUTATION\_FACTOR = 0.975.

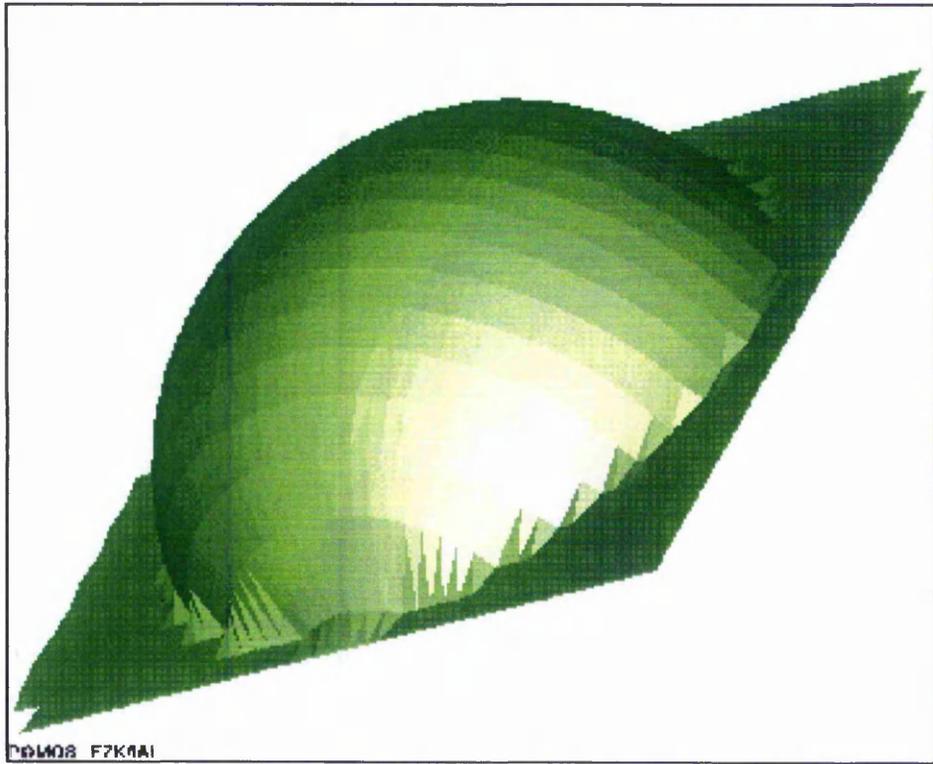
Figure 4.6.1 (b) shows that the surface has been segmented into a planar segment (orange surface part) and a spherical segment (green surface part). Because merely two segments meet which can be clearly distinguished by colour, black boundary curves for each segment have been omitted. Table 4.6.1.1 and Table 4.6.1.2 show the estimated characteristic parameters of the planar and of the spherical segment, respectively.

# $\Delta$	Estimated characteristic parameters			
	Surface normal (x, y, z)			Distance
731	-0.00001	0.00301	1.00000	1.55701
	Optimised characteristic parameters			
	Surface normal (x, y, z)			Distance
731	-0.00060	-0.00054	1.00000	1.14323

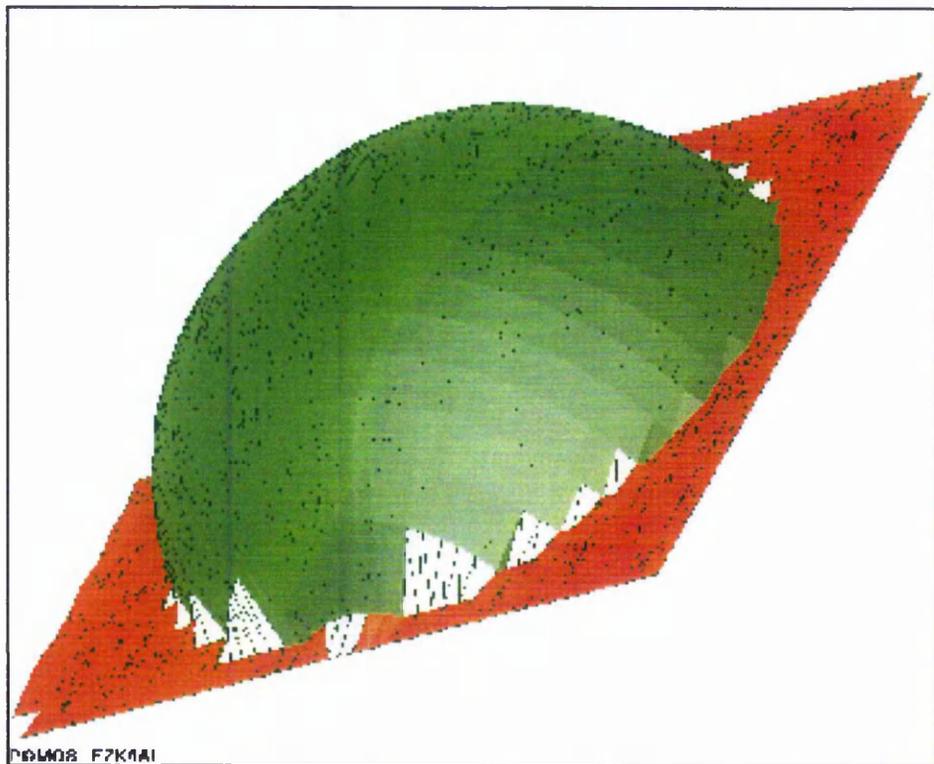
**Table 4.6.1.1:** Parameter estimates for the planar region in Figure 4.6.1 (b)

# $\Delta$	Step ID	Estimated and optimised characteristic parameters			
		Centre (x, y, z)			Radius
171	0	90.59644	111.12756	0.17992	7.60341
	1	90.58913	111.88286	-0.74922	8.86096
	2	90.53621	111.90201	-0.77750	8.77591

**Table 4.6.1.2:** Parameter estimates for the spherical region in Figure 4.6.1 (b) (step ID: 0 = initial estimate, 1 = output of GA, 2 = output of Direct Search optimisation).



(a)



(b)

**Figure 4.6.1:** A segmented triangulated surface of a hemisphere attached to a plane (2178 triangles; 1159 points; x, y, and z dimensions in units: approx.  $20.5 \times 19.5 \times 9.5$ )

- (a) Rotated view showing the composed character of the surface
- (b) Extracted planar (orange) and spherical segment (green); characteristic parameters of both are given in Tables 4.6.1.1 and 4.6.1.2. Processing time: approx. 01:16:26 hours

## 4.6.2 Segmentation of a mechanical part

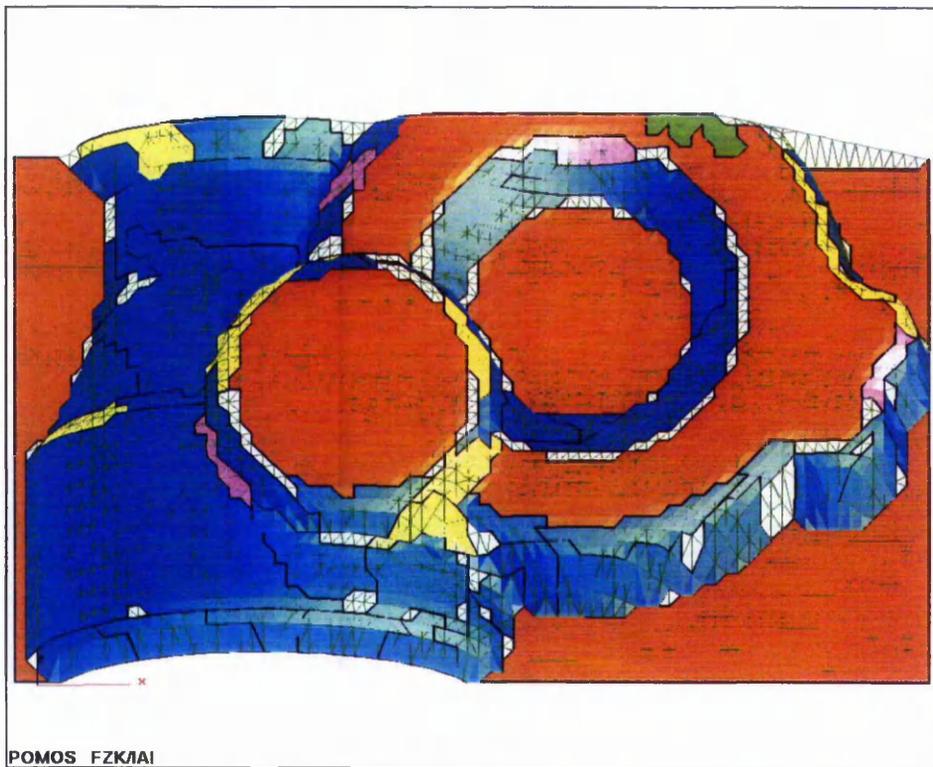
The data of the surface in Figure 4.6.2 has been triangulated with the POMOS software system (the original point data can be downloaded from an FTP site listed in [Hoschek 96] and is likely obtained from parts of a car engine). Details about the measurement accuracy of the data are unknown. Characteristic parameters for the cone have been determined according to method A. A tolerance value  $\tau$  automatically deduced from the data amounts to 0.13015.

Further settings of algorithmic parameters:

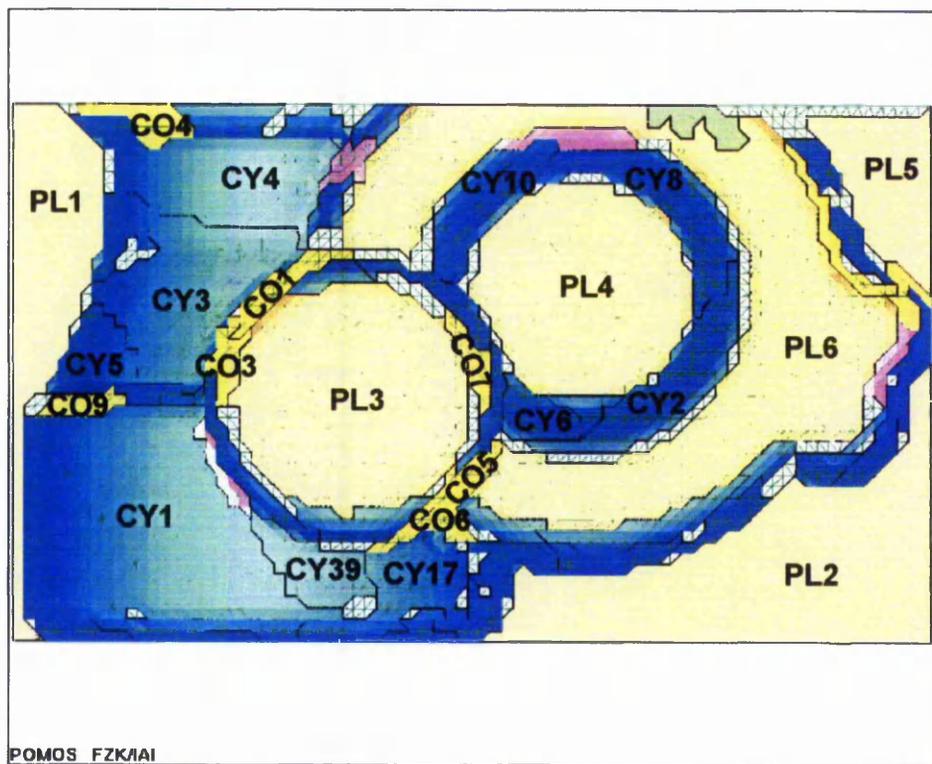
- MIN\_TRIANGLES = 20,
- GENERATIONS = 1000, and
- MUTATION\_FACTOR = 1.0 (i.e. no reduction has been applied).

The extraordinary large value for GENERATIONS explains the slow algorithmic performance.

The surface has been segmented into 6 planar, 1 spherical, 42 cylindrical, 9 conical, and 4 toroidal regions. Table 4.6.2.1 shows estimated characteristic parameters for all planar segments; Tables 4.6.2.2 and 4.6.2.3 do the same for a selection of large cylindrical and conical segments, respectively.



(a)



(b)

**Figure 4.6.2:** *The triangulated surface of a mechanical part of Figure 3.5.1*  
 (a) *Tilted view showing the composed character of the surface.*  
 (b) *Extracted regions are labelled as listed in Tables 4.6.2.1 - 4.6.2.3. Processing time: approx. 81:38:30 hours.*

Plane ID	# $\Delta$	Estimated characteristic parameters			
		Surface normal (x, y, z)			Distance
PL1	382	0.00101	0.00402	0.99999	0.98500
PL2	808	-0.00102	-0.00204	1.00000	0.90883
PL3	674	0.03057	0.01651	0.99940	28.23091
PL4	586	0.00594	0.01280	0.99990	12.64053
PL5	211	0.00098	-0.00388	0.99999	0.87997
PL6	1146	0.01488	0.02417	0.99960	20.63088
		Optimised characteristic parameters			
		Surface normal (x, y, z)			Distance
PL1	382	0.00101	0.00402	0.99999	0.98500
PL2	808	-0.00102	-0.00204	1.00000	0.90883
PL3	674	-0.01256	0.02542	0.99960	27.02864
PL4	586	-0.00914	0.01979	0.99976	12.13028
PL5	211	0.00098	-0.00388	0.99999	0.87997
PL6	1146	0.00305	0.02333	0.99972	20.06453

**Table 4.6.2.1:** *Estimates of characteristic parameters for the planar regions in Figure 4.6.2*

Prim. ID & # $\Delta$	St. ID	Estimated and optimised characteristic parameters						
		Reference point (x, y, z)			Axis (x, y, z)			Radius
CY1, 723	0	18.99055	10.54519	2.79157	-0.01966	0.99944	-0.02690	19.04895
	1	19.61014	-4.08992	2.08589	-0.01660	0.99944	-0.03146	20.04120
	2	19.61014	-4.08992	2.08589	-0.01660	0.99944	-0.03146	20.04120
CY2, 161	0	48.42851	31.33257	22.62965	0.35208	-0.37092	0.85934	13.67776
	1	48.66809	30.93600	27.16639	0.35201	-0.37125	0.85924	15.75447
	2	48.43783	30.94043	27.13757	0.35214	-0.37103	0.85926	15.90282
CY3, 322	0	19.63381	27.46400	1.78857	-0.01502	0.91640	-0.39998	14.55680
	1	19.65662	27.25763	2.07803	-0.03344	0.91539	-0.39992	14.60293
	2	19.65662	27.58670	2.07803	-0.03344	0.91593	-0.39992	14.60293
CY4, 276	0	20.07911	42.08276	0.86981	-0.00329	0.99988	-0.01486	13.32512
	1	20.04957	66.63225	0.96232	-0.00730	0.99989	-0.01491	13.34698
	2	20.32250	64.91354	0.88232	-0.00713	0.99989	-0.01491	13.33739
CY5, 81	0	19.26922	18.46411	1.65235	-0.44544	-0.89336	-0.05904	17.58528
	1	18.46536	18.09524	1.14454	-0.44526	-0.89338	-0.06013	17.59901
	2	18.47202	17.54839	1.68410	-0.44527	-0.89337	-0.06013	17.59893
CY6, 54	0	47.23732	29.88938	21.31810	0.32316	0.39072	-0.86192	11.78000
	1	46.81442	29.88623	31.17702	0.32300	0.39066	-0.86195	16.55107
	2	46.81539	29.88623	31.17229	0.32296	0.39057	-0.86195	16.55601
CY8, 165	0	48.28821	31.78440	20.57710	-0.40873	-0.32461	-0.85297	12.77392
	1	48.41047	31.73452	20.45490	-0.40918	-0.32537	-0.85253	12.75025
	2	48.28333	31.74665	20.45537	-0.40892	-0.32539	-0.85253	12.75122
CY10, 123	0	46.87870	32.88107	22.79690	0.37539	-0.36593	-0.85157	12.69199
	1	46.58829	32.92215	22.85442	0.37571	-0.36736	-0.85076	12.66719
	2	46.57393	32.79846	22.84641	0.37564	-0.36728	-0.85079	12.74563
CY17, 108	0	26.17038	7.22515	7.67526	0.03288	0.99903	-0.02936	12.12349
	1	26.15922	5.69852	7.80014	-0.00552	0.99917	0.03993	12.38450
	2	25.98973	6.13369	7.78258	-0.00017	0.99909	0.04255	12.33787
CY39, 94	0	19.98116	4.99644	0.42040	0.42448	-0.90497	-0.02906	21.83376
	1	19.50878	6.27048	0.63603	0.42060	-0.90489	-0.06531	21.71992
	2	19.50827	7.04463	0.63594	0.42070	-0.90488	-0.06472	21.71978

**Table 4.6.2.2:** Estimates of characteristic parameters for 10 selected cylindrical regions (out of 42 in total) in Figure 4.6.2 at different steps of the parameter optimisation (St. ID: 0 = initial estimate, 1 = output of GA, 2 = output of Direct Search optimisation)

Prim. ID & # $\Delta$	St. ID	Estimated and optimised characteristic parameters						
		Apex (x, y, z)			Axis (x, y, z)			Op. angle
CO1, 39	0	21.56125	31.52365	25.30011	0.51688	-0.81190	-0.27140	87.20166
	1	21.65438	31.50527	22.85449	0.50995	-0.81295	-0.28123	7.84822
	2	21.65438	31.50535	23.87022	0.50972	-0.81310	-0.28110	7.84750
CO3, 30	0	16.48872	24.46872	25.13436	0.88029	-0.37103	-0.29566	86.80119
	1	13.88644	22.31338	17.49718	0.88051	-0.37035	-0.29567	86.83897
	2	13.66237	21.61134	18.76907	0.88023	-0.37123	-0.29578	86.91202
CO4, 41	0	11.44541	46.58763	14.44755	0.41359	0.72937	-0.54495	86.89075
	1	10.46850	45.63629	12.90319	0.41397	0.72924	-0.54471	74.02611
	2	10.46915	45.56146	13.08385	0.41348	0.72924	-0.54516	75.03278
CO5, 32	0	37.55233	14.67548	25.63850	-0.77765	0.50450	-0.37516	87.57207
	1	38.21827	14.98532	23.28163	-0.77359	0.50363	-0.38440	79.64618
	2	38.22299	14.98541	24.70762	-0.77359	0.50363	-0.38462	79.65943
CO6, 28	0	32.98512	10.11816	21.84942	0.35261	-0.86765	0.35049	83.91502
	1	33.63419	8.79293	16.43088	0.35333	-0.87659	0.32670	6.76917
	2	33.62177	9.09183	16.41765	0.35330	-0.87659	0.32669	6.93362
CO7, 28	0	39.70795	24.37162	25.15377	-0.91334	-0.30184	-0.27332	86.84280
	1	39.31090	24.60444	23.89106	-0.91236	-0.30439	-0.27389	8.83344
	2	39.31063	24.69303	23.98498	-0.91244	-0.30473	-0.27311	8.64772
CO9, 25	0	7.45638	21.31503	16.47953	0.50667	-0.69939	-0.50412	87.37781
	1	7.92412	27.16567	5.27306	0.53066	-0.71789	-0.45059	85.70634
	2	7.46628	24.31649	10.57388	0.53066	-0.71789	-0.45059	85.70723

**Table 4.6.2.3:** Estimates of characteristic parameters for 7 selected conical regions (out of 9 in total) in Figure 4.6.2 at different steps of the parameter optimisation (St. ID: 0 = initial estimate, 1 = output of GA, 2 = output of Direct Search optimisation); the last column indicates the opening angle in degrees.

### 4.6.3 Segmentation of a watering can

Differing from the standard extraction order, the geometric primitives for this example have been extracted in the order plane, sphere, cone, cylinder, and torus. Characteristic parameters for the cone have been determined according to method B. A tolerance value  $\tau$  automatically deduced from the data amounts to 3.022484. Further settings of algorithmic parameters are:

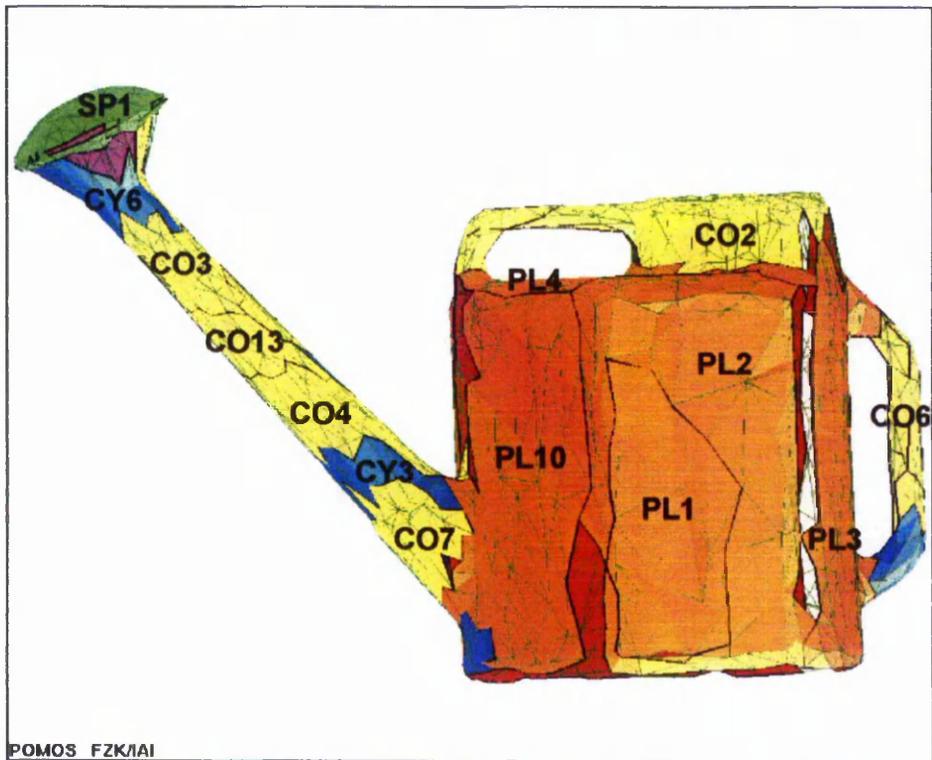
- MIN\_TRIANGLES = 20,
- GENERATIONS = 1000, and
- MUTATION\_FACTOR = 0.997.

Regions corresponding to the geometric primitives that have been extracted by the segmenter are presented in Figures 4.6.3.1 - 4.6.3.3 and in Tables 4.6.3.1 - 4.6.3.5. In total, 10 planar, 1 spherical, 14 conical, 6 cylindrical, and 1 toroidal surface segments have been extracted.

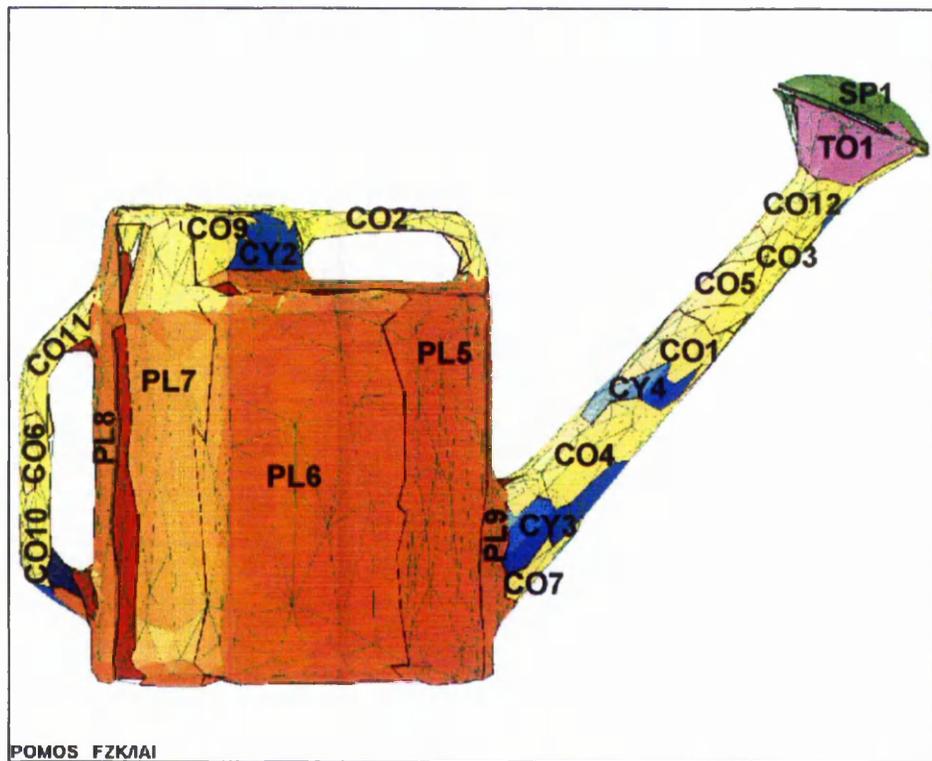


POMOS FZK/AI

**Figure 4.6.3.1:** *A triangulated watering can made up of 3000 triangles and 1556 points (x, y, and z dimensions in units:  $530.360 \times 351.880 \times 235.180$ ).*



(a)

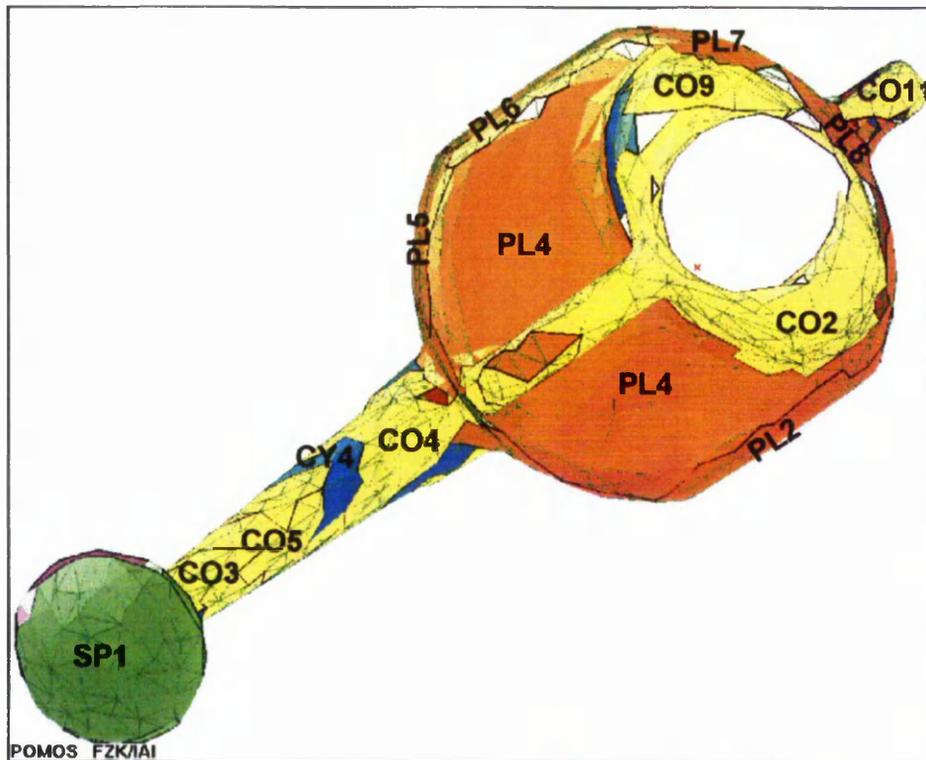


(b)

**Figure 4.6.3.2:** *The segmented triangulated watering can of Figure 4.6.3.1. Identified regions are labelled in accordance with Tables 4.6.3.1 - 4.6.3.5.*

*(a) View from left*

*(b) View from right*



**Figure 4.6.3.3:** The segmented triangulated watering can of Figure 4.6.3.1 viewed from the top. Identified regions are labelled in accordance with Tables 4.6.3.1 - 4.6.3.5

Plane ID	# $\Delta$	Estimated characteristic parameters			
		Surface normal (x, y, z)			Distance
PL1	42	0.19760	0.02629	0.97993	3.022484
PL2	148	0.27948	0.00774	0.96012	108.43168
PL3	177	0.95469	0.02736	0.29635	142.14079
PL4	224	0.00679	0.99979	0.01918	72.36367
PL5	134	-0.82782	-0.05210	-0.55857	93.64571
PL6	163	-0.34384	-0.01478	-0.93891	105.33434
PL7	82	0.43390	0.01157	-0.90089	138.23586
PL8	127	0.99160	0.04675	-0.12058	146.21342
PL9	93	-0.97591	-0.04212	-0.21405	85.40946
PL10	153	-0.77518	0.00782	0.63169	81.72922
		Optimised characteristic parameters			
		Surface normal (x, y, z)			Distance
PL1	42	0.19508	0.01895	0.98061	102.93008
PL2	148	0.24114	0.03187	0.96997	98.51686
PL3	177	0.96383	0.00551	0.26645	142.27626
PL4	224	-0.02970	0.99914	0.02897	72.25000
PL5	134	-0.82940	-0.04054	-0.55718	88.90900
PL6	163	-0.20166	-0.03560	-0.97881	104.23564
PL7	82	0.52165	0.02427	-0.85282	138.58667
PL8	127	0.96911	0.04593	-0.24231	146.69077
PL9	93	-0.97991	-0.03094	-0.19704	84.59544
PL10	153	-0.69457	-0.00170	0.71943	84.11395

**Table 4.6.3.1:** Estimates of characteristic parameters for the planar regions in Figures 4.6.3.2 and 4.6.3.3

Prim. ID	# $\Delta$	Step ID	Estimated and optimised characteristic parameters			
			Centre (x, y, z)			Radius
SP1	171	0	-276.95944	116.95547	57.78614	67.89391
		1	-239.07851	15.88060	22.73441	169.82486
		2	-237.42599	9.52883	27.46443	175.31607

**Table 4.6.3.2:** Estimates of characteristic parameters for the spherical region in Figures 4.6.3.2 and 4.6.3.3 (Step ID: 0 = initial estimate, 1 = output of GA, 2 = output of Direct Search optimisation).

Prim. ID & # $\Delta$	St. ID	Estimated and optimised characteristic parameters						
		Apex (x, y, z)			Axis (x, y, z)			Op. angle
CO2, 350	0	93.69340	81.67758	27.96159	0.02205	0.99762	-0.06541	-
	1	-1106.50610	3.26102	-261.99481	-0.16484	0.92165	0.35104	89.43
	2	-1095.94861	2.69610	-259.56747	-0.16484	0.92165	0.35104	89.44
CO4, 132	0	-136.83072	-23.89913	-14.25910	-0.07107	-0.05662	0.99586	-
	1	-1312.02283	-200.98459	-418.17996	0.06005	0.86460	-0.49829	0.56
	2	-1311.31482	-199.09320	-418.17957	0.06005	0.86460	-0.49839	0.54
CO7, 65	0	-97.28217	-71.25775	21.76221	-0.16955	-0.37663	0.91071	-
	1	-1560.21948	-347.97174	-715.45294	0.12728	0.83179	-0.53933	89.59
	2	-1555.12744	-338.33850	-698.05640	0.12728	0.83179	-0.53933	89.59
CO12, 60	0	-270.46786	92.89755	15.80743	0.33951	0.03973	0.93976	-
	1	-2513.37256	-1741.60901	-1594.42065	-0.09063	0.19733	-0.97564	25.53
	2	-2475.97266	-1741.60901	-1582.73950	-0.09063	0.19733	-0.97564	25.53

**Table 4.6.3.3:** Estimates of characteristic parameters for 4 selected conical regions (out of 14 in total) in Figures 4.6.3.2 and 4.6.3.3 at different steps of the optimisation (St. ID: 0 = initial estimate, 1 = output of GA, 2 = output of Direct Search optimisation); each opening angle of a conical segment is measured in degrees; “-” indicates a numerically indefinite value.

Prim. ID & # $\Delta$	St. ID	Estimated and optimised characteristic parameters						
		Reference point (x, y, z)			Axis (x, y, z)			Radius
CY3, 55	0	-138.87518	-33.69337	36.28129	-0.66720	0.73627	0.11292	16.83415
	1	-140.04120	-22.65806	29.16896	-0.66720	0.73627	0.11367	23.97616
	2	-140.04120	-22.65806	29.16896	-0.66720	0.73627	0.11555	25.38327
CY6, 53	0	-265.10159	113.80539	58.77826	0.67892	-0.72903	-0.08713	21.52509
	1	-267.05453	108.59653	58.76721	0.67706	-0.73030	-0.10135	25.20853
	2	-267.05453	108.59653	58.76721	0.67591	-0.72915	-0.11399	25.03676

**Table 4.6.3.4:** Estimates of characteristic parameters for 2 cylindrical regions (out of 6 in total) in Figures 4.6.3.2 and 4.6.3.3 at different steps of the parameter optimisation (St. ID: 0 = initial estimate, 1 = output of GA, 2 = output of Direct Search optimisation).

Characteristic parameter	Initial estimate	Output of GA	Output of DSO
Centre of gravity (x)	-2.25437	-263.53055	-263.40326
(y)	-0.16425	-102.34827	-102.34962
(z)	1.05999	-271.09650	-292.67905
Main plain normal (x)	-0.15572	-0.10948	-0.10948
(y)	-0.87090	-0.87870	-0.87870
(z)	-0.46613	-0.46450	-0.46450
Major radius	361.05072	131.44841	107.67413
Minor radius	8.05160	397.00476	412.08682

**Table 4.6.3.5:** *Estimates of characteristic parameters for the toroidal region TO1 (encompassing 54 triangles) in Figure 4.6.3.2 at different steps of the parameter optimisation (GA = Genetic Algorithm, DSO = Direct Search Optimisation)*

#### 4.6.4 Segmentation of a toy boat

Figure 4.6.4.1 illustrates the triangulated surface of the un-segmented data set of a toy boat. Instances of geometric primitives have been extracted from this surface in standard order. A tolerance value  $\tau$  automatically deduced from the data according to Section 5.2.2 amounts to 0.001136. Further algorithmic parameters have been set to:

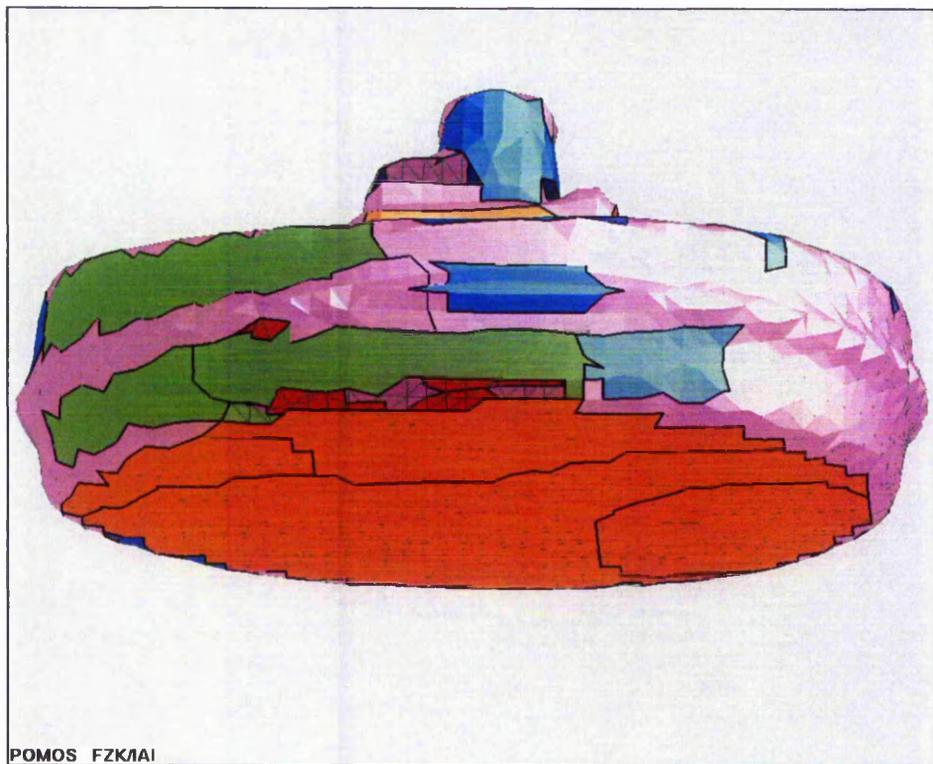
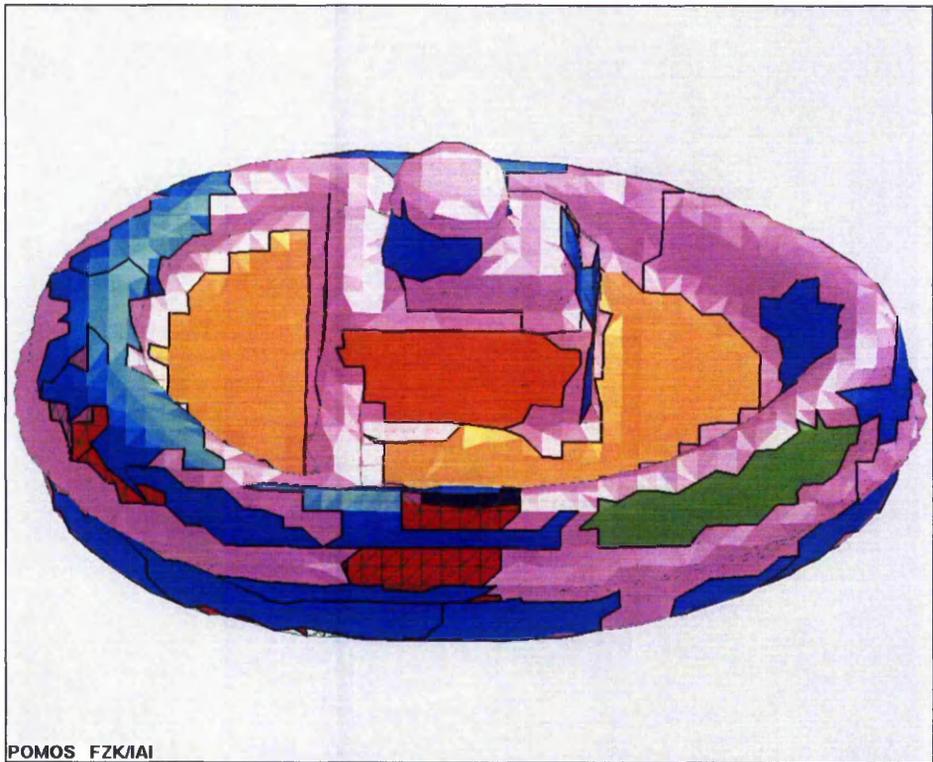
- MIN\_TRIANGLES = 20,
- GENERATIONS = 300,
- MUTATION\_FACTOR = 0.99.

The extracted surface segments are shown from different views in Figures 4.6.4.2 - 4.6.4.4. Owing to shortcomings in image region labelling no characteristic parameter values will be given. As a result the surface of the toy boat has been segmented into 9 planar, 4 spherical, 18 cylindrical, 0 conical, and 5 toroidal segments in total. On the other hand, the extraction of the toroidal segments appears to be erroneous. Nevertheless, the segmenter achieved a surface decomposition into regular triangulated surface parts.



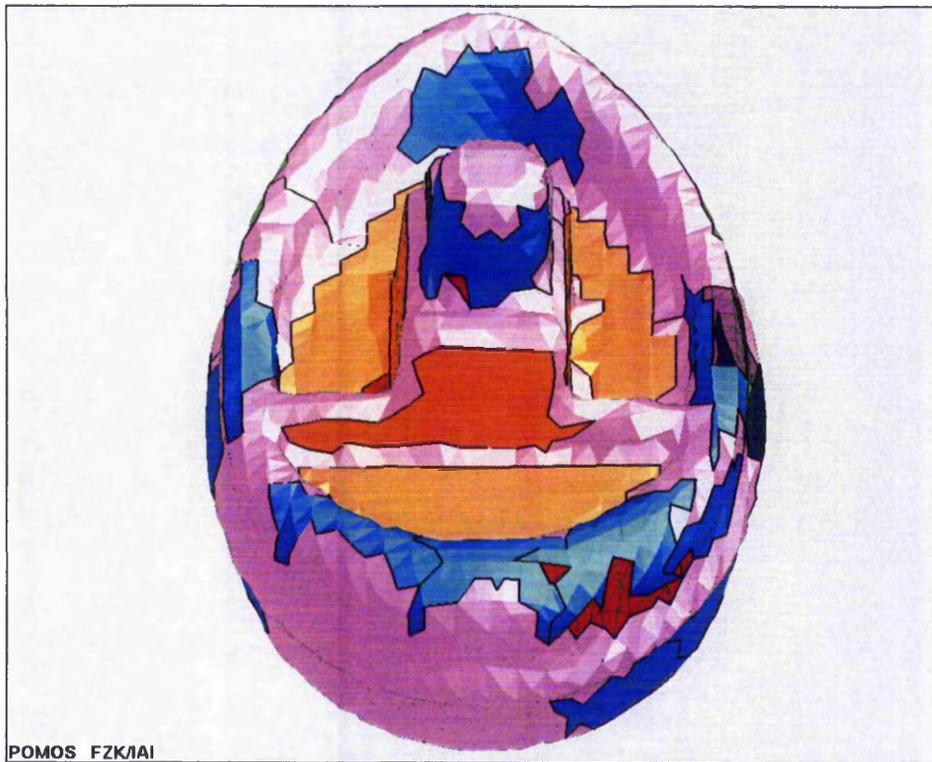
POMOS FZK/IAI

**Figure 4.6.4.1:** *A triangulated toy boat made up of approx. 15500 triangles and approx. 7900 points (x, y, and z dimensions in units:  $0.389 \times 0.695 \times 0.386$ ).*

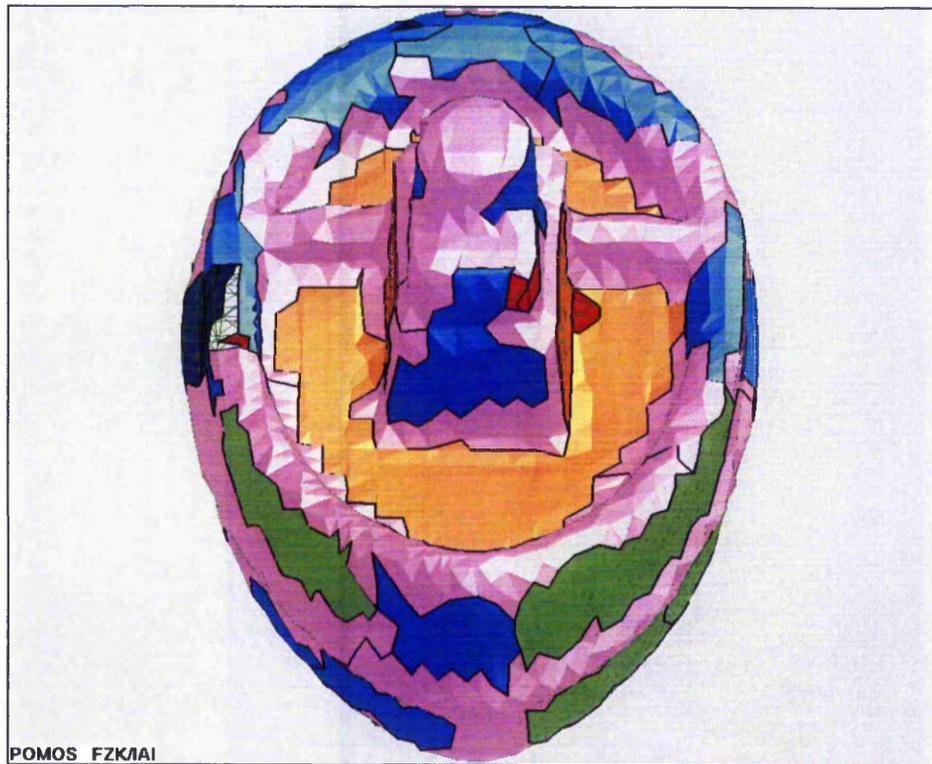


**Figure 4.6.4.2:** *The segmented toy boat of Figure 4.6.4.1 viewed from two different positions. Processing time: approx. 2:31:55 hours. Identified regions are not labelled but are colour encoded (as described in the preface of Section 4.6).*

(a) *View from top right*  
(b) *View from lower left*



(a)



(b)

**Figure 4.6.4.3:** *The segmented toy boat of Figures 4.6.4.1 and 4.6.4.2 viewed from a further two positions.*

(a) *View from top rear*

(b) *View from top front*

## **4.7 Discussion of methods and results**

The previous sections have presented a specific approach how to segment a triangulated surface. This section intends to justify the algorithmic details and to reflect on the level of segmentation achieved. In particular, the extraction of geometric primitives by the modified region growing algorithm in Section 4.2 is discussed in Section 4.7.1, and the estimation of the characteristic parameters in Section 4.7.2. The sensitivity of the algorithmic parameters to small changes is subject of Section 4.7.3, whereas the segmentation results of Section 4.6 are assessed in Section 4.7.4.

### **4.7.1 Extraction of geometric primitives by modified region growing**

For successive region growing one might argue that the order in which geometric primitives are extracted has such an impact on the result that this order requires careful consideration. However, the question whether an “adequate extraction order” exists at all requires further investigation. Preliminary results for successive extraction suggest that the “optimal” extraction order needs to be determined for each triangulated surface individually. Many of the existing segmenters obey an extraction order imposed by increasing length of the vector of characteristic parameters. For this reason the order in this thesis has been mostly selected as planes, spheres, cylinders, cones, tori, and has been specified different otherwise.

The result of the above successive segmentation algorithm is a regular triangulated surface segmented into regions each of which is tagged with its type of geometric primitive, its corresponding characteristic parameters, and its boundaries. On the other hand, a surface segmentation in general does not need to be unique, as some vertices near region boundaries may belong to more than one geometric primitive. So alternatively segmentation by region growing can be performed concurrently, where concurrent means that a triangle can be tagged with more than one type of geometric primitive. The more recent segmentation approaches seem to favour concurrent labelling obtained from parallel processing. For successive segmentation it is expected that the output segments will strongly

depend on the order the geometric primitives are extracted (besides other factors such as data accuracy and allowed tolerance). For concurrent segmentation a triangle can be added to a segment to which it fits best, i.e. where the distance criterion for its vertices yields a minimum. So the decision to which segment the triangle belongs can be corrected, if necessary. This makes the need for a particular extraction order obsolete. In any case, these topics require further investigation.

#### 4.7.2 Estimation of characteristic parameters

In general, estimates using compensated centres  $C^*_i$  of triangles are afflicted with the inaccuracy that these points do not lie on the surface of a geometric primitive but inside if the primitive is convex and outside if it is concave. Such an inaccuracy is expected to have only little effect on, for example, the estimation of the apex of a cone as the geometric calculations are based on lines between two  $C^*_i$  that are shifted either a small distance into or out of the surface. This is true for the estimates of the characteristic parameters of the cone in Section 4.3.4 and of the torus in Section 4.3.5 where  $C^*_1$ ,  $C^*_2$ , and  $C'_2$  are all below the surface.

In case of the cone one could argue that the estimates for  $C'_2$  would be more precise if  $\delta_{1, \max}$  were replaced by  $\delta_{2, \max}$ . However, in this case the approximate projection of  $C^*_2$  into the plane  $Z_1$  could be located *above* the cone's surface, which in turn implies an estimate for the apex further away from both points as the opening angle  $2\alpha$  would be more acute. Similarly, the estimates of the characteristic parameters for the torus could be affected.

It is noted that method A for estimating the characteristic parameters of a cone yields more robust results than method B. This is expected to result from a numerical inaccuracy in Equation (4.15) because the estimate for the opening angle depends on  $C_{\text{con}}$ , which can be considered as an estimate of "lower degree" as it is not directly deduced from measured quantities. On the contrary, in method A the opening angle depends only on quantities that involve less error-prone calculations.

A more recent alternative and likely more reliable method for the estimation of characteristic parameters involves the calculation a point  $\mathbf{P}$  on the axis of a cylinder, a cone, or a torus by

$$\mathbf{P} = \mathbf{C}^* + R_{\max} \mathbf{n}^* \quad (4.36)$$

where  $\mathbf{C}^*$  denotes the compensated centre of a triangle,  $\mathbf{n}^*$  its associated compensated normal, and  $R_{\max}$  the radius associated with the principal curvature value  $\kappa_{\max}$ . Two of such points can then be used to estimate  $\mathbf{a}_{\text{cyl}}$ ,  $\mathbf{a}_{\text{con}}$ , or  $\mathbf{a}_{\text{tor}}$ , respectively, so that the need for the computation of approximate projections of a compensated centre can be bypassed. This also avoids the use of principal curvature directions as they are only the best out of the 9 possible ones available, which is likely to improve the parameter estimation. A more detailed description of this method for cylinder, cone, and torus can be found in appendix C.

### 4.7.3 Robustness of the algorithmic parameters

The settings of the algorithmic parameters introduced in Section 4.5 that guide the segmentation process have been based on heuristics rather than systematically. However, the settings for the majority of these parameters are not sensitive in the sense that slight changes of the values do not cause significantly alter the segmentation results. For most of them there exists a specific interval of robustness. Further details about the relevance and the robustness of the algorithmic parameters are explained below according to the Authors' judgement. The settings of the values for most of the presented algorithmic parameters may have some potential for further improvements. A better tuning of the algorithmic parameters remains, however, a multi-dimensional optimisation problem in itself that is difficult and time-consuming to solve.

Now for each algorithmic parameter the relevance is discussed and, where applicable and known, its individual interval of robustness is explained. Firstly, DIAMETER is a data-dependent parameter that is sensitive insofar as it controls the algorithmic parameters CURVATURE,  $\tau$ , MUTATION\_VARIANCE, and STEP\_WIDTH. The purpose of DIAMETER is to provide a (linear or reciprocal) scaling factor for these parameters. Because it is directly deduced from the data,

an interactive user-defined setting of this parameter is not required, and hence an interval of robustness cannot be given.

CURVATURE, a parameter reciprocally linked to DIAMETER, serves as a threshold that allows the distinction between planar and curved regions during region growing. If the value for CURVATURE were set too high, then triangles in curved regions would be classified as good seeds for the growing of planar regions. Correspondingly, if the value were set too low, then triangles in nearly flat regions would be considered as good seeds for the growing of geometric primitives that have a bent surface. However, setting this parameter to  $2.0 / \text{DIAMETER}$  in this thesis turns out as not too sensitive since the segmentation results are rather affected by region growing than by seed determination (not shown here).

The next two algorithmic parameters can be considered as the most sensitive ones since a small change of their values can alter the segmentation result significantly. If the value of the tolerance parameter  $\tau$  is set too high, then, for example, a triangulated cylindrical surface will be decomposed into stripes of planes. On the contrary, if its value is set too low, no region will grow large enough to pass the growth validation test where only regions having more triangles than  $\text{MIN\_TRIANGLES}$  are classified as valid regions. Because a “noisy” (i.e. inaccurate) scan may produce a “noisy” triangulated surface, the setting of  $\tau$  is preferably deduced from the data. One way to do this may be to fit a small plane (or another geometric primitive) to the data points and to determine the average point-to-plane distance. To some extent this distance measures the amount of noise in the data, so that for a setting of  $\tau$  to three times this distance may well allow for a sufficient growth of regions. This is what has effectively been done in section 5: since planar regions are found in a majority of surface scans taken for engineering purposes an automated tolerance deduction as suggested makes sense. Because during region growing planes are grown first (in fact the tolerance value  $\tau$  is obtained from the first plane of estimated low curvature that has been grown in the data where no tolerance value is used for the first 10 triangles of the region, say),  $\tau$  is available for the expected correct classification of all regions in a triangulated surface. However, if the surface does not have any nearly flat region

then the tolerance may be determined by an initial fit using higher order primitives. In the unlikely case that no primitive can be fitted to the triangulated surface, it may be discarded as irregular.

Since the relevance of MIN\_TRIANGLES has already been explained in the above paragraph, it remains to state the difficulties to find a proper setting of this value and a possible interval of robustness. In the worst case a triangulated surface could consist of only a few large triangles, so that it would make little sense to infer a proper setting of MIN\_TRIANGLES from such a quantity as total number of surface triangles. It is suggested by the Author to involve statistic quantities (e.g. the average triangle area) to improve the setting of MIN\_TRIANGLES. Particularly in curved surface parts a larger number of triangles per surface area unit can be expected (especially when the surface is pre-processed by a triangle compression algorithm), so that it might be worthwhile to adjust the algorithmic parameter value according to the estimated surface curvature. For these reasons an interval of robustness cannot be stated in general.

A group of six algorithmic parameters controls the behaviour of the genetic algorithm employed for optimising the characteristic parameters of a geometric primitive during region growing: GENERATIONS, MUTATION\_RATE, MUTATION\_VARIANCE, MUTATION\_FACTOR, POPULATION\_SIZE, and BEST\_NUM. Setting the number of generations too large, beyond 300, say, forces the algorithm to perform slower with only little adjustment of the characteristic parameters to the desired optimal values. So this parameter is fairly robust as has been demonstrated in the first segmentation example in section 4.6.1, where it has been set to 1000. For small numbers of GENERATIONS the optimisation may prematurely end in a local rather than a global minimum so that region growing can lead to an over-segmented surface.

GENERATIONS is closely related to POPULATION\_SIZE, the number of individuals employed for the optimisation by the GA per generation. Setting the value of the latter parameter too high obviously slows down the algorithmic performance. On the other hand, a too small value does not provide a sufficiently large number of guesses of characteristic parameters that are required to find the

global optimum. So reasonable values for GENERATIONS are likely to be found in the interval [30, 50].

As MUTATION\_RATE controls the rate of individuals in the population whose genes are modified in each generation, a value of 0 for this parameter obviously does not perform any optimisation at all. On the contrary, a value of 1 causes a change of genes for all individuals in a generation that is not desirable when the adjusted characteristic parameters are already near the optimal solution. Thus a value in between of 0.3, say, provides a proper setting of this parameter.

MUTATION\_VARIANCE, the measure of variance in a gene of an individual, is a rather uncritical algorithmic parameter provided that it is set “sufficiently” larger than 0. A too small variance hinders the GA to generate the genetic diversity needed to converge to the global minimum, whereas a too large variance slows down the algorithmic performance. However, the setting  $0.75 * \text{DIAMETER}$  used here seems to work well.

MUTATION\_FACTOR induces an elemented of iterated expansion or contraction of the interval of mutation variance. Because of this, a value exceeding 1 is likely to deteriorate the performance of the GA owing to an expansion of the genetic diversity, and a value much smaller than 1 leads to a quick contraction. Since the interval of mutation variance is multiplied by MUTATION\_FACTOR to the power of GENERATIONS, a reasonable setting for MUTATION\_FACTOR is  $(0.001 * \text{MUTATION\_VARIANCE})^{1 / \text{GENERATIONS}}$ , say. This setting limits the interval of mutation variance to 0.1 percent of the initial interval size.

Again the computational time increases if the POPULATION\_SIZE of the GA is set to high, beyond 50, say. A size smaller than about 20 has approximately the same effect as in this case only a small number of individuals with “good” genes are likely to emerge. Therefore in this thesis it is suggested to use values in the range of 30 and 50, a range that has been found not to be critical.

The larger BEST\_NUM, the number of the best individuals per generation, the more individuals survive in the next generation, and the less potential exists for the remaining  $\text{POPULATION\_SIZE} - \text{BEST\_NUM}$  individuals to form a genetic

pool for further improvements on the current (in general non-optimal) solution. Though this number is not too sensitive to changes, a value in the range of 0.2 to 0.5 multiplied by POPULATION\_SIZE seems to be adequate.

The roles of the algorithmic parameters STEP\_WIDTH and STEP\_FACTOR used for the Direct Search optimisation resemble very much the parameters MUTATION\_VARIANCE and MUTATION\_FACTOR for the GA, respectively. STEP\_FACTOR controls the contraction of the search space around the neighbourhood of a point, and STEP\_WIDTH refers to its initial dimension. Both values are insensitive for settings that are similar to those suggested in section 4.5.

Owing to complexity some of the above algorithmic parameters may have not been set to optimal values. However, determining an optimal setting for all algorithmic parameters forms a much larger optimisation problem than the one discussed here and is beyond the scope of this thesis. It therefore remains part of future work.

#### **4.7.4 Evaluation of the segmentation results**

For a proper evaluation of the quality of a segmenter it is desirable to have something like a “gold standard” available. A “gold standard” can be thought of a data set containing parts of all types of geometric primitives such that joints between all possible pairs of types of geometric primitives occur. Furthermore, this standard needs to possess well-defined measurement accuracy and to allow for a unique segmentation.

If the problem to segment the triangulated surfaces that have been used in this thesis were given to a large number of engineers, it would be unclear whether they could agree on a unique solution for each of the surfaces. Such a solution could be considered as the ideal standard segmentation to which the result of an automated segmenter could be compared with.

However, such a standard is not available (if it exists at all), and thus a rigorous assessment of the results produced by the segmenter that has been developed for

this project cannot be given. Instead, the results are assessed using a less objective methodology, namely by visual inspection and by comparing the expected to the effectively achieved surface decomposition.

In general, the above segmenter tends to perform over-segmentation and, apart from the toroidal segment in Example 4.6.4, almost no under-segmentation. No noise segments have been produced, and not very many triangles at the borders of valid surface segments have been missed. In principle, the output of missed segments has been expected, as some surface parts (especially of high curvature) are too small in order to form valid segments.

Next, it is briefly commented on the segmentation results shown in Section 4.6. The surface of the hemisphere attached to a plane in example 4.6.1 has been decomposed fairly well apart from missed triangles at the common border of planar and spherical segment in Figure 4.6.1 (b). The latter phenomenon is likely to arise from a non-optimal triangulation. Presumably this can be corrected by re-triangulation as is discussed in Section 6.

The segmentation of the surface of the mechanical part in Figure 4.6.2.2 has also given a good result. Over-segmentation mainly occurs in curved surface areas where a number of small segments of various types of geometric primitives can be found. The number of planar segments agrees with the number that is expected by a human expert although the boundary curves of the segments sometimes appear to be jagged. Several cylindrical segments have also been extracted well. However, a conical patch on the left-hand side of the surface has been misclassified as a cylindrical segment. The same seems to be the case for cylindrical regions around the planar disk on the right.

The result for the segmentation of the watering can in Figures 4.6.3.2 and 4.6.3.3 are less good, in particular because of misclassifications. For example, it can be observed that the planar segments PL3, PL5, PL7, and PL10 effectively show a cylindrical character. The handle on the top of the can (labelled by CO2) needs to be considered as under-segmented. Furthermore, the long cone-shaped spout between the body and the rose of the can has been significantly over-segmented (although most of the conical segments are correctly classified). On the other

hand, the rose has been segmented fairly well, and most of extracted planar segments show a good correspondence to the expected result. However, the correctness of the extraction of the toroidal segment TO1 underneath the rose is difficult to verify.

Essentially the same applies to the segmentation result of the toy boat in Figures 4.6.4.2 and 4.6.4.3. For spherical and cylindrical segments the result seems to be acceptable, whereas many toroidal segments appear to be under-segmented. The hull of the toy boat shows some missed segments. Planar segments at the bottom of the boat appear to be over-segmented, whereas planar segments on top of the boat have been extracted well. This indicates that either an inappropriate tolerance has been used for the segmentation or (which is believed by the Author) that the measurement errors in the data points are not normally distributed.

## **4.8 Summary**

The segmentation of a triangulated surface into parts of geometric primitives is a delicate task. Because the vertices of a triangulated surface are in practice subject to measurement inaccuracies, the extracted surface segments need to tolerate a certain amount of noise in the data. Consequently the segmentation problem has been theoretically defined in Section 4.1 depending on a tolerance value to substantiate what can be expected as a good segmentation result.

As an appropriate bottom-up approach for the extraction of surfaces, region growing is well established. However, previous region growing algorithms most often relied on parameterised data. For this reason an algorithm for region growing has been established that is suitable for non-parameterised triangulated data. Seed regions for the growing of geometric primitives are placed according to the occurrence of eight fundamental surface types that can be distinguished by using the estimated curvature value of Chapter 3. For these seed regions that consist either of one or two triangles, methods have been developed to determine initial characteristic parameters for each geometric primitive.

During region growing these characteristic parameters are likely to need an adjustment. Each time after a point has been added to the current region that

does not fit to the shape defined by the parameters, it is attempted to adapt them to the whole region. The goodness of fit of the points to the assumed underlying surface is measured as the maximum of all distances between one point and the surface. So the adjustment of characteristic parameters for an instance of a geometric primitive to fit the surface to the points is a process that involves numerical optimisation. The majority of such optimisation methods require a “good initial guess” for the parameters that is then successively refined, and thus the initially determined characteristic parameters were used accordingly.

Because the objective function is non-linear and non-smooth (i.e. without partial derivatives) conventional optimisation methods such as the well-known Levenberg-Marquart method for least-square problems cannot be applied. Moreover, the objective function is likely to possess numerous local minima. Hence a GA has been employed to adjust the parameters close to the global minimum, and then a Direct Search optimisation with better algorithmic performance has been applied for the “fine tuning” of the solution.

The usefulness and the capability of the proposed methods have been demonstrated on some triangulated surfaces that are affected by noise in the data points. Furthermore, the initially and the finally determined characteristic parameters for each type of geometric primitive have been presented in order to allow an evaluation of the segmentation results. For the identification of planar patches on a triangulated surface a special technique has been developed; therefore the extraction of planes is explained in more detail in Chapter 5. However, because an engineer is likely to produce a different segmentation result, further improvements on the segmentation of a triangulated surface are likely to be necessary.

## 5 Improved extraction of planar segments

Planes play an important role in design and engineering. They can be found, for example, as surface elements of casings and foam elements inserted into boxes that save transport goods. Therefore, the extraction of planar segments deserves particular attention. In Chapter 4 a method based on numerical optimisation has been presented to extract parts of each of the geometric primitives considered in this thesis (i.e. plane, sphere, cylinder, cone, and torus) from a triangulated surface. However, the extraction of planes employing numerical optimisation is not very fast.

Thus, this chapter is especially dedicated to an extraction of planar segments from a given set  $D$  of data points. The geometric method presented here employs the same modified region growing algorithm for the extraction of planar segments as described in Figure 4.2.1, but it is significantly faster because it requires no numerical optimisation. It requires information about the connectivity of the points in  $D$  only (e.g. if  $D$  is a triangulated surface, the vertices are edge-connected). This makes the method applicable to a wide range of data structures such as octrees and voxel sets.

In a nutshell, the proposed approach is based on the idea of associating with every planar region a **reference triangle** (RT) and an **expanded triangle** (ET). Both triangles are initialised by three non-collinear points of  $D$ , henceforth referred to as **triplet**. The same triplet is also selected as a seed region.

The key idea of the method is to abandon the time-consuming fitting of data points to the RT as often as possible by making a plane available (represented by the ET) that approximates the current region better, if required. So always after a data point has been added to the current region, the RT is retained unless the point does not fit to the plane represented by the RT. In the latter case, the triplet of the RT is replaced by the one of the ET. Each time after a “growth step” of the current region (i.e. after a point has been added), the ET is updated to the triplet of points in the current region spanning the largest area. So the RT is updated less often than the ET. This technique to monotonically expand the area of the ET is expected to compensate for measurement errors in the data. Thus, at any stage the area of the ET is equal or larger than the one of the RT and represents a plane

that approximates the current region “the best”. On the other hand, the RT represents at any stage a plane that approximates the current region “sufficiently well”.

Extracting planar segments is described in the following sections in more detail, where the basic update scheme for the ET is introduced and problems associated with this approach are pointed out. Section 5.2 presents a remedy to these problems to make the technique fast and robust for the extraction of planes consisting of three parts. Section 5.2.1 describes how to use the local fitting of a plane to analyse the measurement errors of the points in  $D$ . Section 5.2.2 explains how to obtain an adaptive threshold from this analysis for the extraction of planes. In Section 5.2.3 the benefit of selecting seed triplets pseudo-randomly is discussed. Thereafter Section 5.3 shows examples of extracted planar segments, and a summary in Section 5.4 concludes this chapter.

## 5.1 The basic method and its problems

This section explains the basic method for the fast extraction of planes based on geometric principles and investigates when this methods fails.

Initially, a seed triplet is selected as a seed region  $\mathbf{R}$  serving as an initial planar segment which also initialises the reference triangle RT and the expanded triangle ET. Both triangles play an important role during region growing. The plane represented by the RT (more precisely: the plane through the triplet of the RT) fits at any stage “sufficiently well” to the current region, i.e. no point in this region has a distance to the plane exceeding a predefined tolerance  $\tau > 0$ . The ET represents a plane of a potentially more accurate fit which is required if during a region growing step a point cannot be added to the current region. Ideally, the seed triplet is of estimated low (mean) curvature because such a triplet is expected to be located in a flat surface part where  $\mathbf{R}$  can “grow well”; later on in this section the case is discussed in which no curvature information is available.

Initial estimates for the characteristic parameters  $\mathbf{n}$  and  $d$  determining the plane through the triplet of the RT uniquely can be gained easily as follows:  $\mathbf{n}$  is taken as the unit vector perpendicular to the RT, and  $d$  is calculated as the smallest distance of this plane to the coordinate origin.

The above plane has an implicit representation as the set of zeros of the function

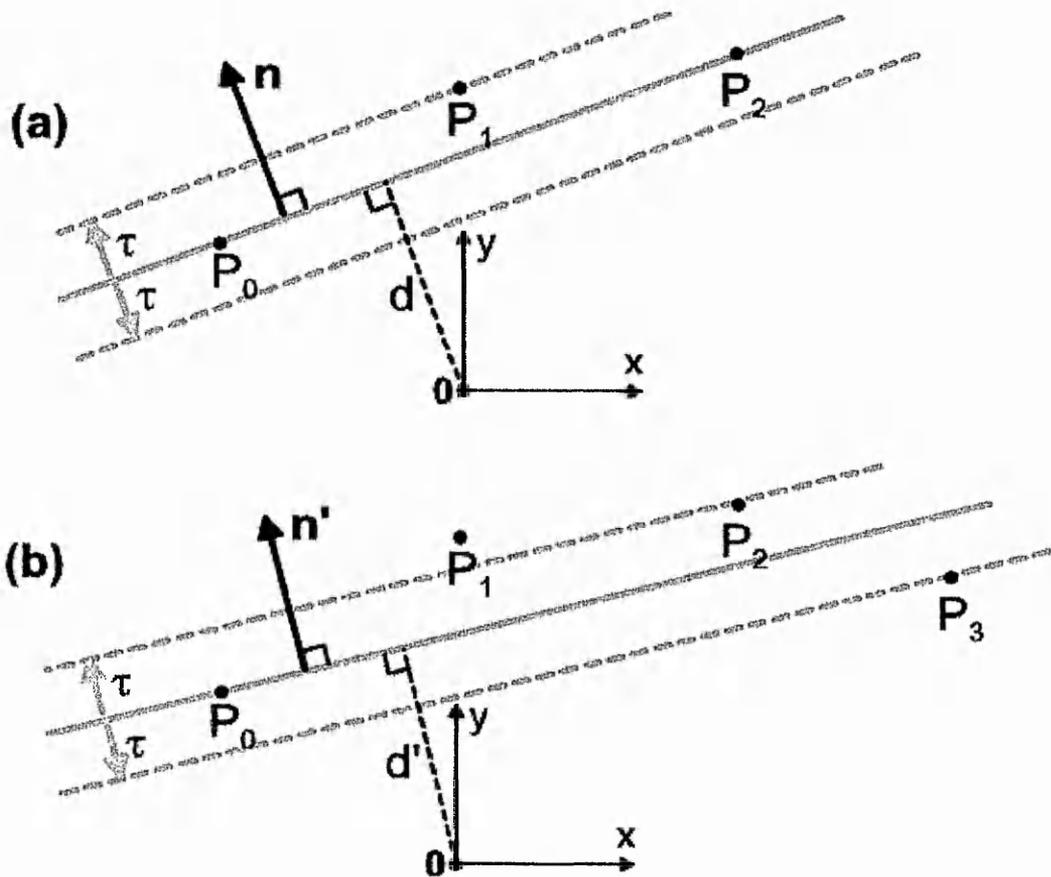
$$f(\mathbf{P}; (\mathbf{n}, d)) = | \langle \mathbf{n}, \mathbf{P} \rangle - d |, \quad (5.1)$$

i.e.  $\{\mathbf{P} \in \mathcal{R}^3 : f(\mathbf{P}; (\mathbf{n}, d)) = 0\}$ . Moreover,  $f$  models the “goodness of fit” of a data point  $\mathbf{P}$  to the plane represented by the RT in the sense that  $f(\mathbf{P}; (\mathbf{n}, d))$  amounts to the distance of  $\mathbf{P}$  to this plane. If, for example, the given data structure is a triangulated surface, and if in Step 2.A.d.iv of the modified region growing algorithm in Figure 4.2.1 a vertex  $\mathbf{V}$  satisfies

$$f(\mathbf{V}; (\mathbf{n}, d)) \leq \tau \quad (5.2)$$

for a preset tolerance  $\tau > 0$ , the triangle associated with  $\mathbf{V}$  is included to the current planar region and is therefore tagged with “TagOfGeoPrim(PLANE)”.

In favour of speeding up the extraction of planar segments from  $D$  the key idea consists in evaluating Equation 5.1 only if necessary. Especially a time-consuming evaluation for multiple points can be necessary during region growing as is shown in the following example. Consider the configuration of points  $\mathbf{P}_0$ ,  $\mathbf{P}_1$ , and  $\mathbf{P}_2$  in Figure 5.1.1 (a) where  $f(\mathbf{P}_j; (\mathbf{n}, d)) \leq \tau$  for a given  $\tau > 0$  and for  $0 \leq j \leq 2$ . Because in Figure 5.1.1 (b) the point  $\mathbf{P}_3$  cannot be added to the current region without modification of  $\mathbf{n}$  and  $d$  an *attempt* is made to adjust these characteristic parameters. The parameters are intended to characterise a plane that fits within tolerance  $\tau$  to all points in  $D$  (in Chapter 4 such an adjustment is achieved by numerical optimisation). After this adjustment the fit needs to be validated for *all* data points because some may have left the tolerance range. As this is true for the point  $\mathbf{P}_1$  in Figure 5.1.1 (b) the validation fails, and thus the point  $\mathbf{P}_3$  is not added to the current region. Abandoning the need to evaluate the goodness of fit in Equation 5.1 for a large number of points can significantly reduce the computational time for the extraction of planes.



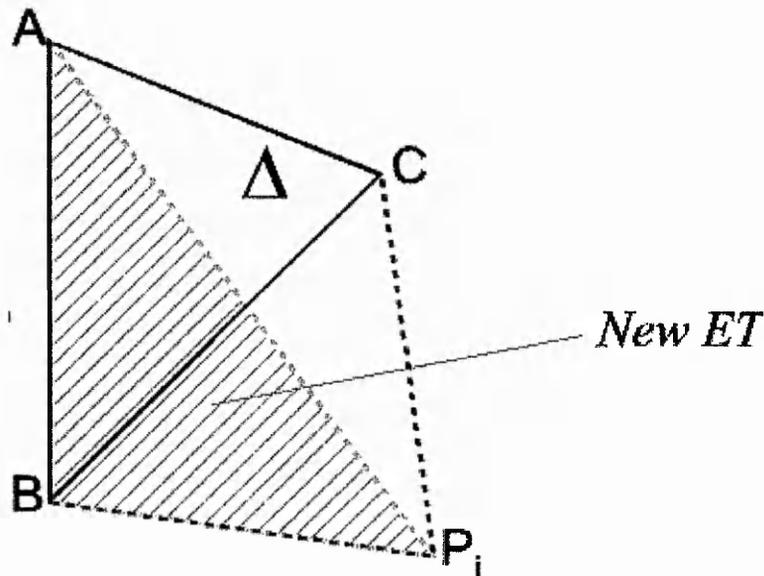
**Figure 5.1.1:** Example for the necessity to check all data points after adding a point to a current planar region (cross-section view).

- (a) The current planar region (grey solid line) given by  $\mathbf{n}$  and  $d$  encompasses the points  $P_0$ ,  $P_1$ , and  $P_2$
- (b) After an update the current planar region (grey solid line) given by  $\mathbf{n}'$  and  $d'$  encompasses the points  $P_0$ ,  $P_2$ , and  $P_3$ , but  $P_1$  has left the tolerance range

Therefore the present work uses a more efficient technique to determining  $\mathbf{n}$  and  $d$  rather than by numerical optimisation: the RT is simply replaced by a triangle of larger area, the ET. In general, this triangle is augmented after each growth step of the current region. This in turn is expected to provide more accurate characteristic parameters for the plane describing the current region (represented by the RT) as it better compensates for measurement errors in the data.

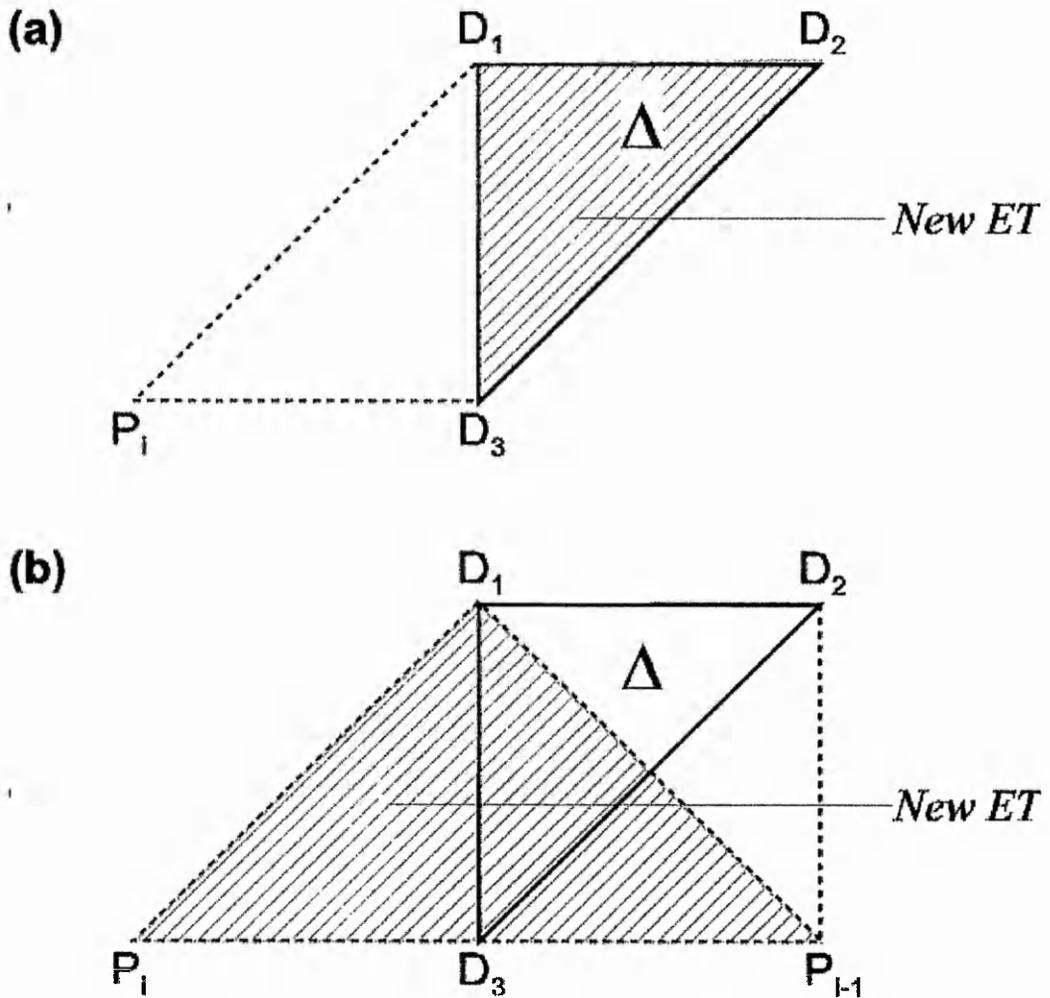
The technique for augmenting the ET during region growing is explained next. Let  $P_i$  be a vertex of a triangle  $T$  adjacent to the current region  $R$  such that the edge of  $T$  opposite to  $P_i$  belongs to the border of  $R$  and  $P_i$  and  $R$  are disjoint (such as  $R = \Delta$  in Figure 5.1.2). Moreover, let  $P_i$  be within the tolerance of  $R$  such that

$f(\mathbf{P}_i; (\mathbf{n}, \mathbf{d})) \leq \tau$ , and let  $\mathbf{P}_i$  be added to  $\mathbf{R}$  after a growth step. In this situation the current ET is updated as follows: all possible triplets out of the four vertices, those formed by the ET together with  $\mathbf{P}_i$ , are examined. Then one triplet spanning a triangle of maximum area is selected as the new ET (see Figure 5.1.2). This procedure ensures after each growth step a monotonous increase of the area of the current ET and is referred to as “basic ET updating scheme”.



**Figure 5.1.2:** *New ET (A, B, P<sub>i</sub>) (shaded) obtained from current ET Δ = (A, B, C) with vertices in R after P<sub>i</sub> has been temporarily added to R*

As is demonstrated next the basic ET updating scheme is likely to fail if the data points are aligned to a regular grid. It is recalled that the ET is updated by replacing one of its vertices by another point  $\mathbf{P}_i$  such that the resulting triangle spans a larger area than the current ET. However, such an update may not be possible because the current ET can have an area that is at least as large as the area of every triangle obtained from combining  $\mathbf{P}_i$  with any pair of points of the current ET (see Figure 5.1.3 (a)). Consequently, the current ET is copied to the new ET that does not expand in area. This can occur more than once in consecutive updates of the ET if



**Figure 5.1.3:** *Special case: all data points are aligned to a regular grid where the current ET is represented by  $\Delta = (D_1, D_2, D_3)$ .*

- (a) *The new ET (shaded) is set to  $\Delta$  as it cannot be expanded (no triangle of larger area can be constructed from three of the points  $D_1$ ,  $D_2$ ,  $D_3$ , and  $P_i$ )*
- (b) *The new ET (shaded) is updated to the triplet  $(D_1, P_i, P_{i-1})$  because the point  $P_{i-1}$  is included to construct a triangle of largest area*

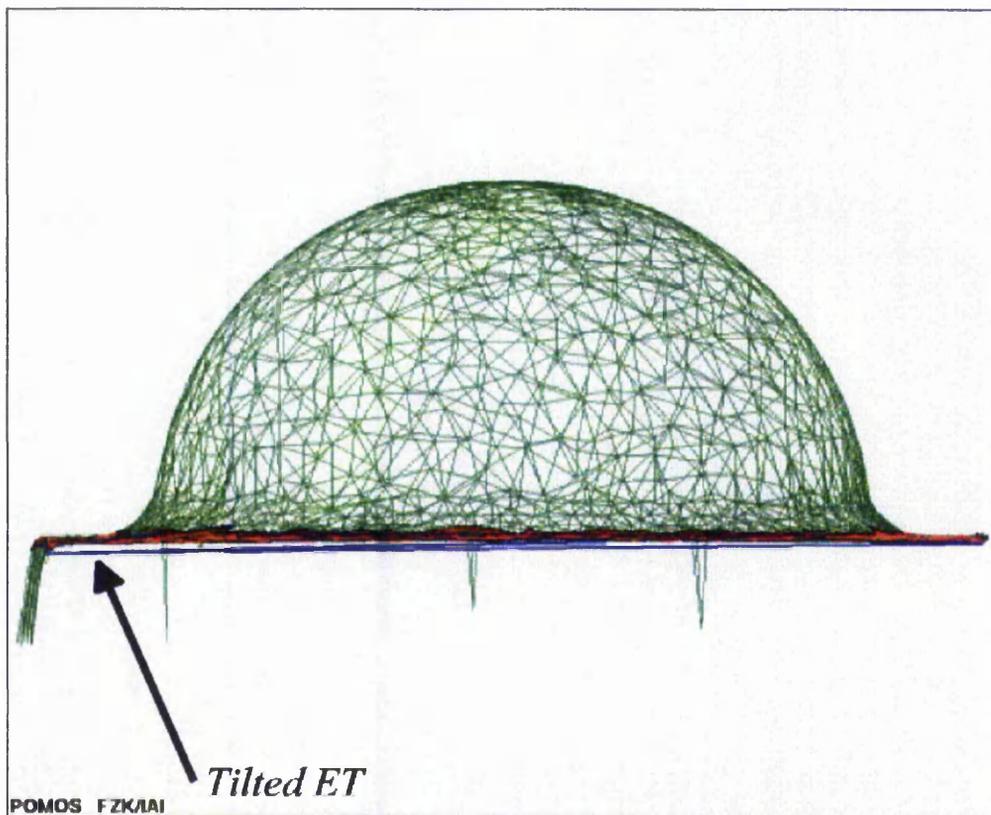
the vertices of D are aligned to a regular grid. In practice it has been observed that the basic ET updating scheme can hinder the ET from growing larger.

As a remedy it is proposed that the new ET should be determined not only from the triplet of the current ET (represented by  $\Delta = (D_1, D_2, D_3)$  in Figure 5.1.3 (b)) and the point  $P_i$  adjacent to  $R$ , but also from the point  $P_{i-1}$  previously added to  $R$ . Merely at the first stage of growing where no  $P_{i-1}$  exists the basic ET updating scheme is applied.

If there is more than one triplet yielding the same area (as it holds for  $(\mathbf{D}_1, \mathbf{P}_i, \mathbf{P}_{i-1})$  and  $(\mathbf{D}_2, \mathbf{P}_{i-1}, \mathbf{P}_i)$  in Figure 5.1.3 (b)) the one with the smallest sum of indices is selected (hence the triplet  $(\mathbf{D}_1, \mathbf{P}_i, \mathbf{P}_{i-1})$  is preferred to  $(\mathbf{D}_2, \mathbf{P}_{i-1}, \mathbf{P}_i)$ ). If no triplet combination yields a larger area than the current ET, then as before the current ET will be retained. Clearly, as the number of data points increases the method will tend to adjust the ET so that it improves the fit of the planar region to the points.

The presented planar extraction technique has been found to perform satisfactory if the seed triangle has estimated low curvature or is located centrally within the planar region to be extracted. However, this assumption can be fairly restrictive since curvature estimates may not always be available. Moreover, the arrangement of triangles on a surface may not allow the selection of an appropriate seed triangle within the centre of a planar region as this region could be long and thin or “U”-shaped. On the contrary, an estimation of curvature prior to plane extraction is likely to consume more computing time than can be saved by an appropriate seed triplet selection; in practice it has been found that the time to avoid “false starts” is outperformed by a “brute force” seed triplet selection though no results are shown here. An improved strategy that can help to overcome this problem consists in a pseudo-random selection of seed triangles (see Section 5.2).

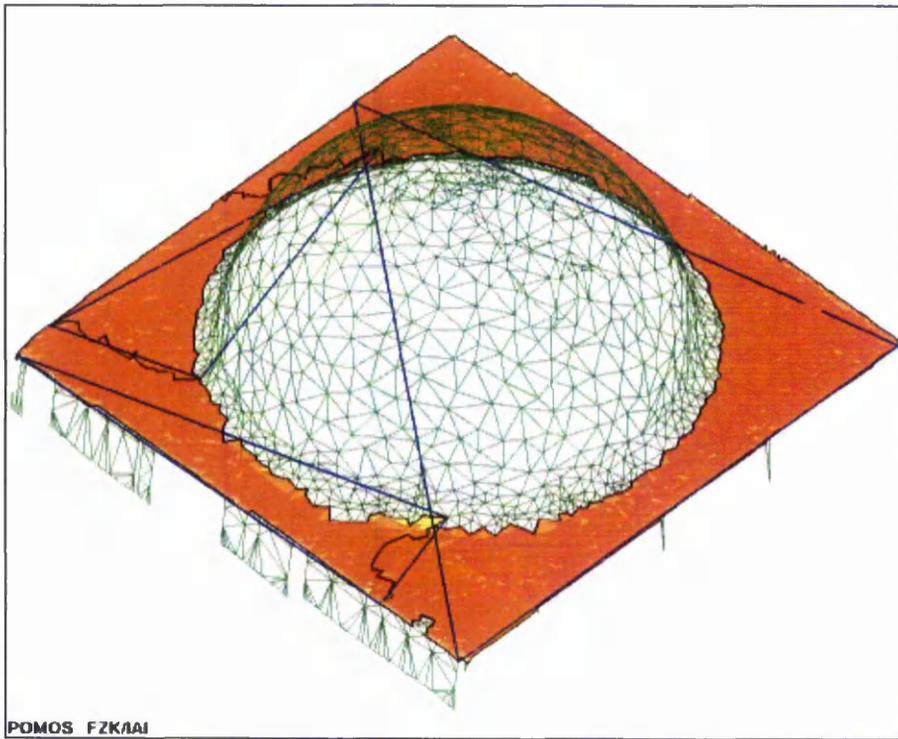
Another crucial element of the presented technique consists in the appropriate selection of the predefined tolerance  $\tau > 0$ . If  $\tau$  is too large, planar regions are likely to grow beyond roof discontinuities or within curved surface parts. On the other hand, if  $\tau$  is too small, planar regions may not grow at all. The first case leads to a “tilt effect” of the ET (illustrated in Figure 5.1.4) impairing the estimates for  $\mathbf{n}$  and  $d$ . This effect can inhibit a planar region from growing sufficiently large. Hence new seed regions are initialised, yielding a split of a potential single planar surface region (the largest of the blue triangles in Figure 5.1.5 corresponds to the “Tilted ET” in Figure 5.1.4).



**Figure 5.1.4:** *The “tilt effect” of the ET (blue lines) occurring in a triangulated surface of a hemisphere on a plane (side view) if the tolerance value  $\tau$  is too large*

Another problem arises when the seed triplet is located in or near a curved area of the surface. A planar region may either be forced to grow merely in a certain direction (e.g. along a surface crease), or growing may be abandoned at all. Here, this problem is denoted as “seed selection problem”. For triangulated surfaces the seed selection problem also appears in a variation of more systematic kind: during the generation of such a surface its facets are imposed a certain order. For example, they may be numbered in ascending order along the surface boundary. The standard ET approach is likely to select a seed triangle near the boundary. This may be unfavourable because measurement inaccuracies in one of the vertices (e.g. in the rightmost point in Figure 5.1.4) cannot be compensated by expansion of the ET any more.

For the above reasons in the following section additional suggestions are made to overcome the identified problems and to enhance the reliability of the method.

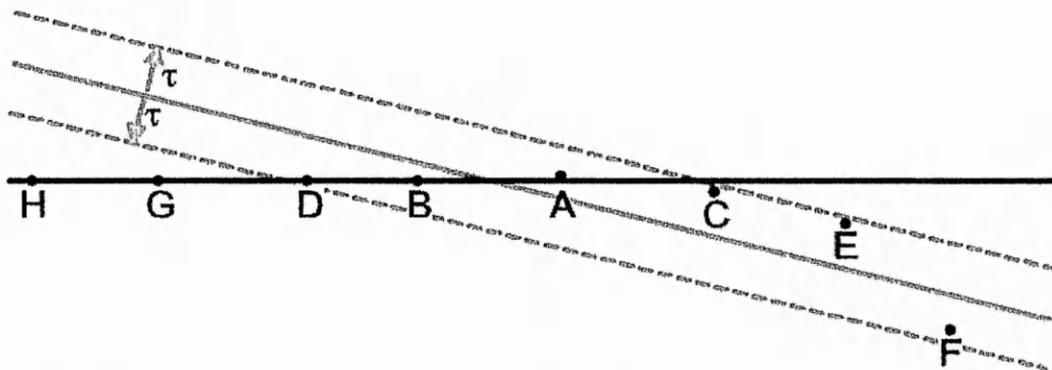


**Figure 5.1.5:** *A “tilt effect” of the first ET (the largest of the above blue triangles) for a too large tolerance  $\tau > 0$  splits the surface in Figure 5.1.4 into three planar segments instead of one (view from above the surface)*

## 5.2 Improvements of the basic method

This section investigates the addressed problems associated with the ET approach of Section 5.1 in further detail and proposes adequate remedies.

Firstly, the problem of the “tilt effect” of the ET is examined. As mentioned earlier this effect can arise from choosing too large a value of the tolerance  $\tau > 0$ : adding a point that is too distant from the current region (such as point **F** in Figure 5.2.1) can lead to this effect.



**Figure 5.2.1:** For a tolerance  $\tau$  too large, the ET (cross-section view, grey line) that represents a planar region encompassing  $\{A, B, C, D\}$  tilts after adding E and F. This prevents G and H from being added though an ET encompassing more vertices exists (cross-section view, black line)

A contribution to remedy this problem may be to analyse the measurement accuracy of D that is subject of Section 5.2.1. The next step explained in Section 5.2.2 is to adjust the tolerance  $\tau$  accordingly; the latter task can be performed during region growing. Another problem is the inappropriate selection of a seed triplet (and thus of the initial ET) that impairs the extraction of planes, and a triplet near a surface boundary is likely to be such one. In Section 5.2.3 a pseudo-random seed selection is proposed as a remedy.

### 5.2.1 Analysis of data accuracy by local fitting of a plane

Next, the possibility of analysing the accuracy of the points in D by fitting a small plane to a region of estimated low mean curvature is discussed.

The analysis is performed as follows: during the extraction of the first plane the average distance  $\bar{a}_m$  of the first m points, say, to this plane is calculated.  $\bar{a} = \bar{a}_m$  serves as an estimate of the order of magnitude of the measurement errors in the points of D.  $\bar{a}$  is henceforth referred to as the **extracted average distance**.

Owing to more accurate sensing during the scanning process the measurement accuracy of the data in flat and low curvature surface parts is expected to be higher than those in more curved regions. This means that  $\bar{a}$  merely allows for coarse error estimation. Therefore a somewhat larger value should be chosen “carefully” as an **adaptive tolerance**  $\tau_a$ , a value gained from analysing the measurement errors of the points in D. Otherwise, if the error is estimated too large, ambiguities in surface segment fitting and the “tilt effect” could arise.

However, the assumption of higher measurement accuracy of the data in sampled planar regions in comparison to curved ones needs to be confirmed by further research in the field of data acquisition.

In this thesis  $m = 10$  has been used to determine the adaptive tolerance  $\tau_a$ , and so the first 10 triangles of the first extracted planar region have been grown unsupervised. By assuming that the measurement errors in the points of  $D$  obey a normal distribution and by demanding that about 99% of these points are fitted to a segment of a plane, a value for the standard deviation of  $\sigma = 2.6\bar{a}$  can be calculated from the extracted average distance. For the extraction of the following planar regions the tolerance  $\tau_a = 3\bar{a}$  has been used as a value that is slightly above  $\sigma$ .

Please note that  $m$  should not exceed the algorithmic parameter `MIN_TRIANGLES`. Otherwise the first plane may have been extracted during region growing without having been able to calculate the extracted average distance (it is recalled that `MIN_TRIANGLES`, introduced in Section 4.5, prescribes the minimum number of triangles required to represent a valid segment of an extracted geometric primitive).

The analysis of the measurement errors in  $D$  has another benefit. A reliable automatic segmentation must answer the question of which data element (this thesis focuses on triangles) needs to be mapped to which data segment. Because this mapping must consider the positions of both, data element *and* data segment, the segmentation also relies on knowledge about the errors in the data affecting these positions. Without this knowledge arising ambiguities in this may complicate the interpretation of a segmentation result, regardless of the types of geometric primitives in which the data may have been decomposed.

## **5.2.2 Extracting planes using an adaptive tolerance**

Next, the implications of replacing the current tolerated distance  $\tau$  by the adaptive tolerance  $\tau_a$  for the growing of planar regions is discussed. As  $\tau_a$  is likely to be smaller than a user-defined tolerance value  $\tau$ , fewer points are added to each current region during region growing. If the seed triplet is located in or near a curved part of the surface, then in general the plane represented by the RT will not

grow very well. So either the region grows in a different direction of estimated low (mean) surface curvature, or the growth terminates prematurely. In the latter case the criterion for a set of points to form a valid segment (in this thesis expressed by MIN\_TRIANGLES) may discard the region as too small. Such a criterion proves to be useful in order to prevent a curved surface such as a sphere of large radius from being fragmented into planar regions, each of which encompasses merely few data points. A new attempt is then made to grow a planar region employing a different seed triplet, possibly in the neighbourhood of the previous one (see next section).

### 5.2.3 Pseudo-random seed triplet selection

As explained before an inappropriate selection of a seed triplet (and thus of the initial ET) cannot be compensated, and a seed triplet near a surface boundary is likely to be such one. Without having estimated curvature values at hand seed triplet are probably selected in some order owing to the underlying data structure - such as a list or a tree. Rather than selecting seed triplets in ascending or descending order, their access in a randomly permuted manner is likely to yield better seed selection results. Such a permutation could be time-consuming to compute, in particular if the number of surface triplet is large. More efficiently, a pseudo-random order can be obtained by considering an arbitrary seed triplet, and then taking every  $m^{\text{th}}$  data point to apply the basic ET updating scheme. Empirically a good choice for  $m$  has been determined by  $m = (3 |D|)^{1/2}$  where  $|D|$  denotes the number of points in  $D$ .

In practical implementations applied to triangulated surfaces the selection of seed triangles indexed in the above order has been found an appropriate substitute for picking triangles randomly. It achieves a reduction of the number of planar regions for otherwise identical settings of algorithmic parameters, in particular MIN\_TRIANGLES, although the result is not optimal.

A single planar region of the surface in Figure 5.1.5 has not been extracted because points of a larger measurement error were likely to be selected as vertices for an update of the ET. It is recalled that so far only attempts have been made to maximise its area. An improved idea consists in restricting the points chosen to update the ET according to their goodness of fit. Empirically, a value of  $1.2\bar{a}$  has

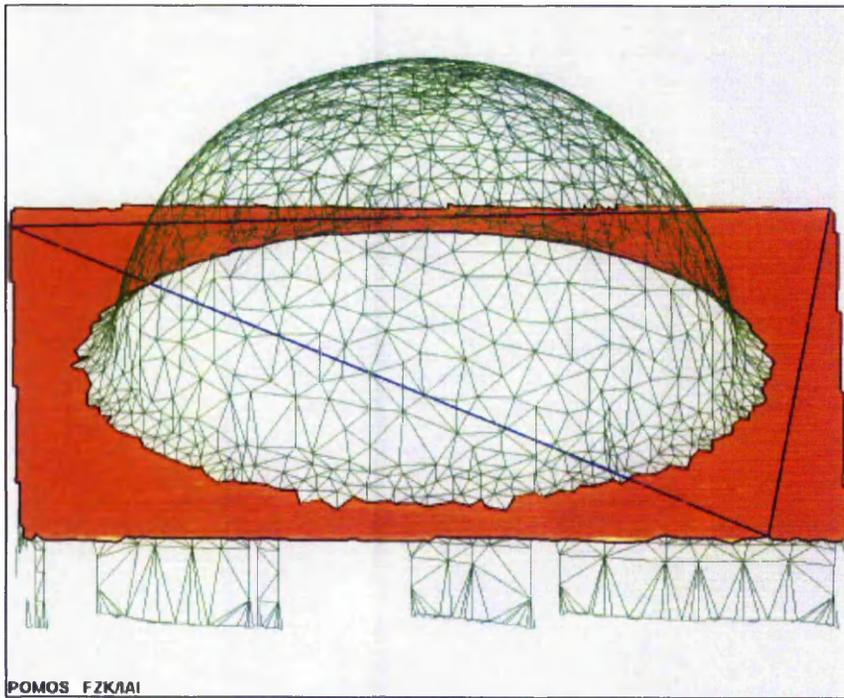
been found a tolerable value for this fit to the current region in order to become an appropriate ET “updating candidate”. Smaller values may prevent the ET from growing larger which may result in a premature termination of growth of the current region, whereas larger values can cause again the “tilt effect” of the ET. Results for the extraction of triangles are shown in the Section 5.3.

### 5.3 Extraction results

Using the improved updating scheme, planar segments have been extracted from the triangulated surfaces presented in Figures 5.3.1 – 5.3.4. The number of points and triangles belonging to this surface is given in Table 5.3.1. Furthermore, the table shows the number of extracted planar segments for each surface, the prescribed minimum number of triangles (where MIN\_TRI abbreviates MIN\_TRIANGLES), and the corresponding computational time.

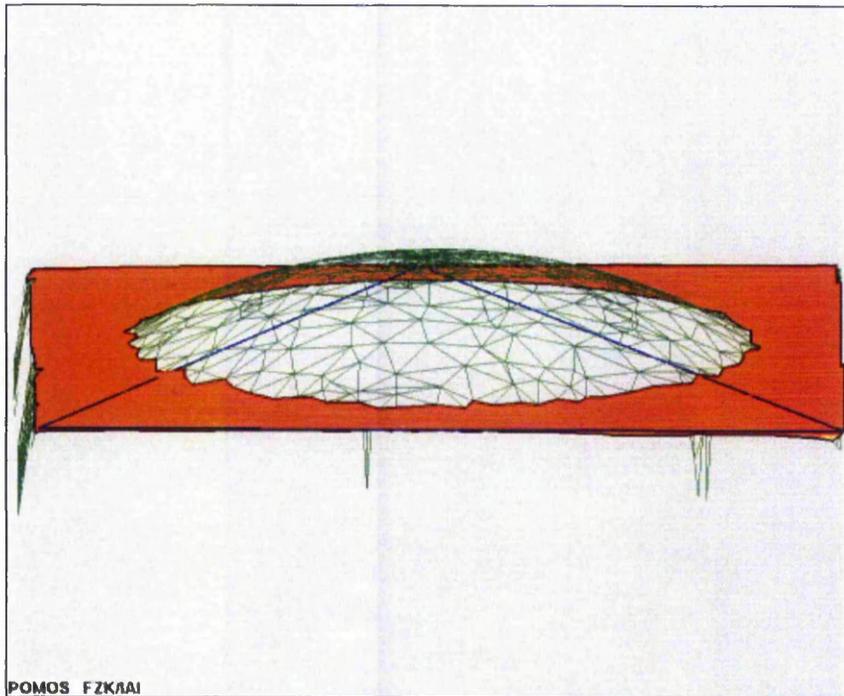
Figure No.	Description	Points	Triangles	MIN_TRI	Extracted regions	Time (secs.)
5.3.1	hemisphere on plane	3473	6646	150	1	13.6
5.3.2	part of ellipsoid on plane	2750	5209	150	1	10.3
5.3.3	technical surface	4018	7776	100	7	24.8
5.3.4	hexagonal nuts	3666	6158	20	20	23.2
5.3.5	keyboard	approx. 11000	approx. 20000	150	6	approx. 11

**Table 5.3.1:** *Overview of the segmentation results presented in Figures 5.3.1 - 5.3.4.*



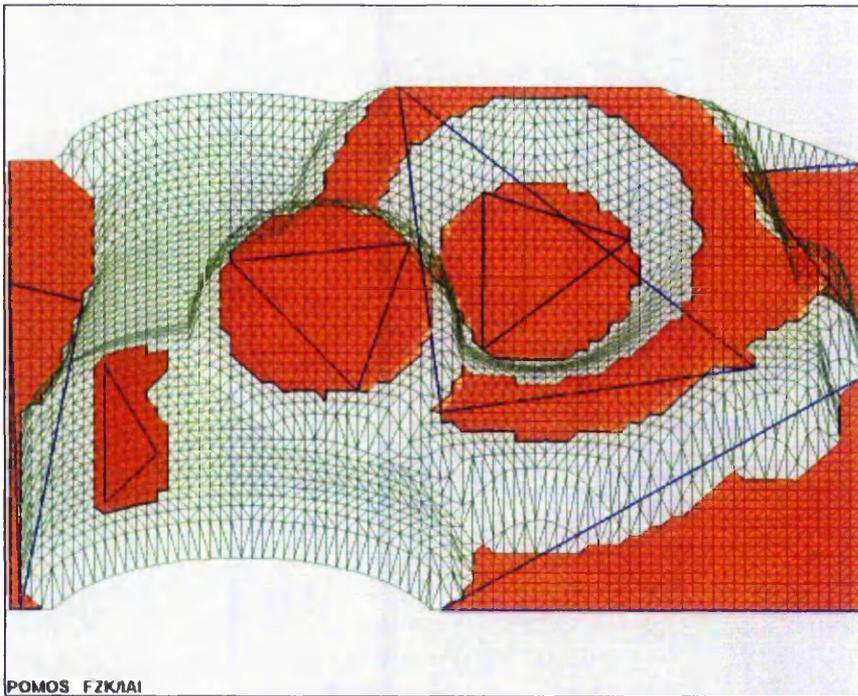
POMOS FZK/IAI

**Figure 5.3.1:** Improved extraction of a single planar segment (orange) with superimposed ET (blue) from the hemisphere on a plane in Figures 5.1.4 – 5.1.5

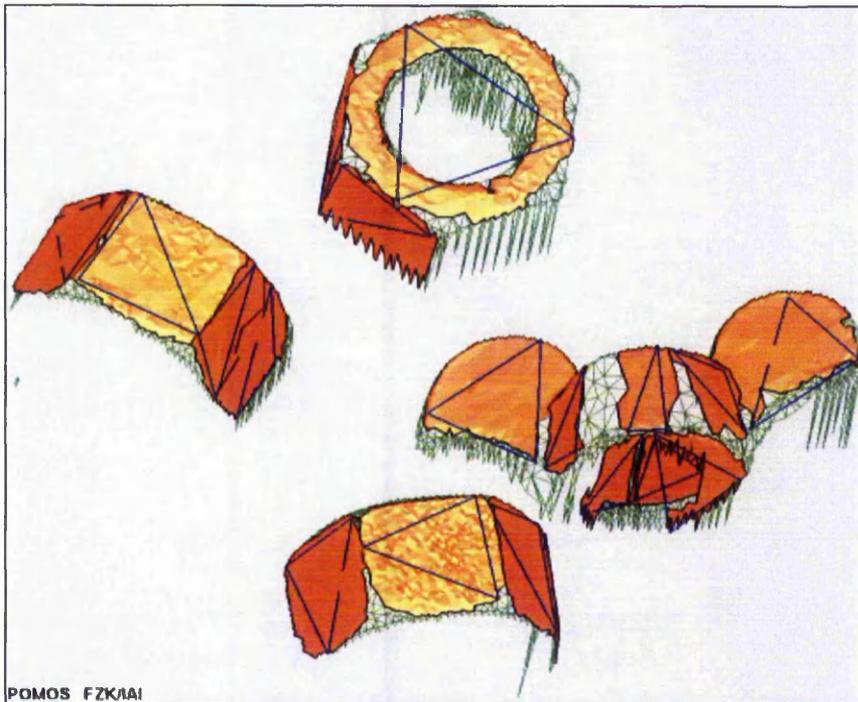


POMOS FZK/IAI

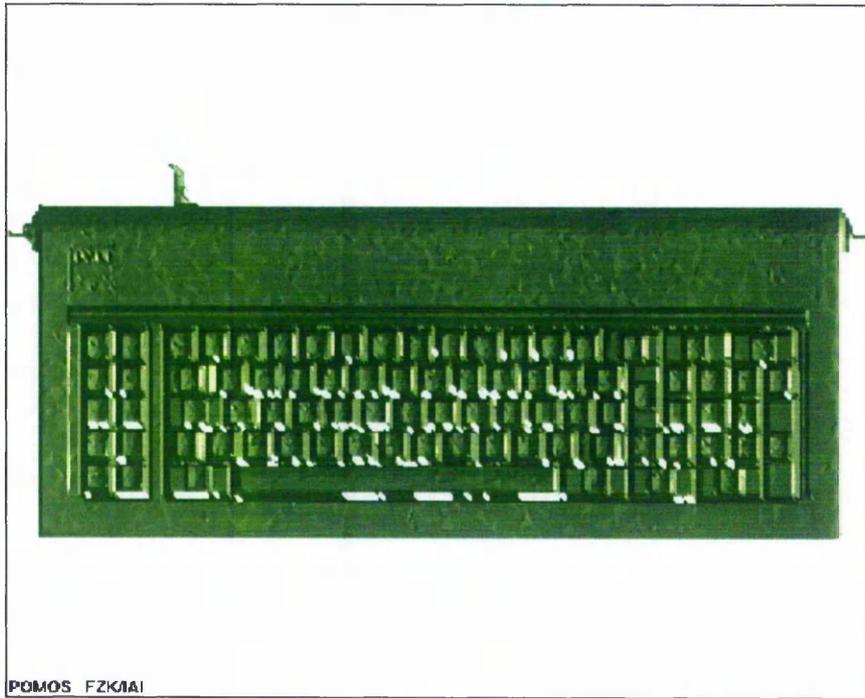
**Figure 5.3.2:** Improved extraction of a single planar segment (orange) with superimposed ET (blue) from a triangulated part of an ellipsoid on a plane



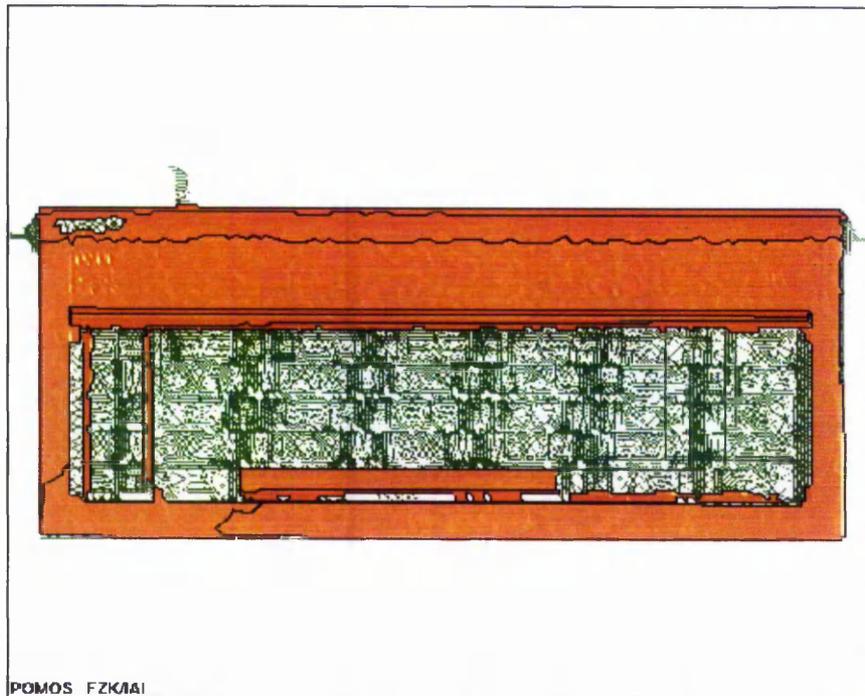
**Figure 5.3.3:** Improved extraction of 7 planar regions (orange with black border) with superimposed ET's (blue) from the triangulated surface of the mechanical part in Figure 3.5.1



**Figure 5.3.4:** Improved extraction of 20 planar regions (orange with black border) with superimposed ET's (blue) from triangulated hexagonal nuts



(a)



(b)

**Figure 5.3.5:** *A triangulated surface of a computer keyboard (top views).*

**(a)** *Shaded surface*

**(b)** *Wire frame representation with 13 extracted planar segments (orange with black border)*

## 5.4 Summary

Naive extraction of planar segments following the approach presented in Chapter 4 is feasible though not efficient. Therefore an improved method has been introduced for the extraction of planar segments from a given set  $D$  of data points. The data does not need to be triangulated but it must be provided with adjacency information. This method exploits the linear nature of planes by employing two especially constructed triangles: a reference triangle (RT) and an expanded triangle (ET). The proposed method successively refines the point-to-plane distance for each of the planar segments to be extracted only if it is necessary in order to gain a good extraction performance.

The above method relies on the deduction of a tolerance value that is used to decide whether a given point in  $D$  belongs to a certain plane or not. Under the reasonable assumption that low curvature surface parts are present in current engineered designs, it is possible to determine an “extracted average distance” while determining the first planar region. Further assuming that measurement error of the data is subject to a normal distribution, an “adaptive tolerance” can be obtained using the extracted average distance. This value guides the subsequent extraction of planar regions. Moreover, extraction is further improved by selecting seed triangles in a pseudo-random manner rather than the order in which they are stored.

The method has shown to be successful and efficient on a number of triangulated surfaces. Since the extraction merely requires adjacency information rather than a surface triangulation, the method can be directly applied to other data structures such as octrees and voxel sets.

## **6 Discussion**

Recovering shape information from a triangulated surface is - in contrast to a relatively easy composition of a surface - a delicate problem. For the determination of parts of geometric primitives and characteristic parameters it may be possible to try other approaches than those presented here. However, the present approach offers a practical solution. It comprises curvature estimation as a preliminary step followed by a phase of iterative region growing. This phase encompasses the initial determination of characteristic parameters for each of the geometric primitives plane, sphere, cylinder, cone, and torus, based solely on geometric calculations using the curvature information. Additionally, it includes the successive fitting of the parameters representing an instance of one of the geometric primitives to a given set of data points, a problem that can be tackled with numerical optimisation. In this thesis the optimisation problem has been formulated in a minimax sense, i.e. minimizing the maximum distances of points in a data set that are part of a homogeneous surface patch; the latter is described by the characteristic parameters. In what follows some advantages and disadvantages of the individual methods and solution components used for the overall segmentation task are discussed and compared to other possible methods. Moreover, topics of future research are outlined.

### **6.1 Curvature estimation**

One might argue that the DN curvature formula for two surface points could be equally well applied to adjacent vertices on a triangulated surface together with their corresponding interpolated normals. Indeed, the curvature estimation phase could be speed up further by such an approach. A possible drawback, however, might be that the directions of principal curvature would be estimated in fewer (discrete) directions than with the NEN method. This is likely to result in a rather inaccurate determination of characteristic parameters for the geometric primitives.

### **6.2 Region growing and determination of characteristic parameters for geometric primitives**

This section discusses the segmentation framework presented in Chapter 4.

### 6.2.1 Segmentation problem definition

Applying the segmentation problem as defined in Section 4.1 only to regular triangulated surfaces is considered as a reasonable restriction. Firstly, the required positive area of each surface triangle avoids pathological triangles that have no area. Secondly, the local parametrisation avoids such cases where triangle edges are shared by more than 2 triangles. Thirdly, surface regularity also includes surface connectivity. For disconnected triangulated surfaces the segmentation problem can be solved by the application of the segmenter to each of its connected surface components individually. By generalising the condition “each subset of a triangulated surface” needs to be of “homogeneous shape” (as in Section 4.1) leads to the condition that “each subset needs to fulfil a specific logical predicate”. Such a generalised problem setting would allow for the segmentation of a surface, for example, into parts of identical colour or homogeneous texture. However, a generalisation of the condition was not necessary for this thesis.

In the definition of the segmentation problem in Section 4.1 a minimal number of segments has been postulated for the decomposition of a triangulated surface into connected and pair-wise disjoint regions of homogeneous shape. Often this problem is defined in a way that maximises the area of each connected extracted region rather than minimising their total number. Both conditions have similar aims, namely to provide a criterion for a segmenter that prevents it from splitting a triangulated surface into a large number of surface parts with similar characteristics.

Although easy to formulate each of these conditions can be difficult to verify in practice. To guarantee a minimal number of segments - depending on a given tolerance  $\tau \geq 0$  - a large number of segmentations (presumably all possible segmentations) needs to be computed and evaluated; this is a major strategic component of the recover-and-select paradigm used by [Leonardis et al. 95]. A benefit of such an approach is that it does not require a precise definition of the size and the boundaries of extracted surface segments. So two segmentation results may be considered qualitatively equivalent if they have the same number of extracted segments, otherwise the result with fewer segments is considered to be better.

Where the appearance of extracted surfaces may remain somehow indefinite for a “minimal number of segments” criterion, the “maximal segment area” criterion offers a better predictability of the segmentation results. The latter criterion depends on the order in which parts of geometric primitives are extracted. Consider, for example, a smooth join between a planar and a cylindrical region. If planes were extracted first, then (of course depending once more on a preset tolerance value  $\tau \geq 0$ ) some triangles of the cylindrical region near the join would likely belong to the extracted plane. Vice versa, if cylinders were extracted first, then the extracted cylindrical region would likely contain some triangles that belong to the planar one. What may actually be needed is a “sensible” balance between the two.

In general it is difficult to compare two different segmentation results. Statistical analysis may be helpful: the results are considered to be equivalent if the extracted segments have approximately the same average area; otherwise the one with the larger average area is deemed to be better. Alternatively, segmentation results may be considered equivalent if the extracted segments encompass approximately the same average number of triangles per segment; otherwise the segmentation with the larger average number of triangles per segment may be regarded as better. In any case, assessing a segmentation result by the number of extracted segments seems to be intuitively a more adequate method.

### **6.2.2 Initial estimation of characteristic parameters for geometric primitives**

To summarise the state-of-the-art in estimating characteristic parameters of geometric primitives from measured data: the problem how to obtain highly accurate estimates from a triangulated surface such that the surface is well approximated by an instance of a specific primitive still requires further investigation. However, it has not been the objective of this thesis to provide a solution to such a delicate problem. It is rather the benefit of this work to contribute methods of estimating characteristic parameters for each of the geometric primitives sphere, cylinder, cone, and torus at all, since only a few methods exist of which the majority are not applicable to triangulated surfaces. The methods presented here can always be applied to initially unstructured data

once a triangulation has been accomplished. It is desirable to investigate and develop alternative methods of determining characteristic parameters of geometric primitives that yield more robust results (i.e. more accurate results in the presence of noise).

For example, the axis of either a cylinder or a cone can be determined by two compensated centres with corresponding normals and the radii of principal curvature given for a pair of (not necessarily adjacent) triangles. This may be achieved because the centre of curvature for principal curvatures of maximum magnitude lies on the axis. The apex of the cone is then a point on the axis at a distance of the above two points that can be computed by the ratio of the principal radii of curvature. These methods differ somewhat from the ones presented in Section 4.3.2, and it is uncertain which of the methods to prefer.

So far it is an open problem whether purely geometrical approaches for the determination of characteristic parameters of geometric primitives exist that do not need any knowledge of second-order differential properties such as surface curvature.

### **6.2.3 Successive parameter optimisation for surface fitting**

In Section 4.4 it has been presented how a surface that is described by a vector of characteristic parameters can be fitted to a given set of data points. The goodness of fit is measured by the maximum distance of a data point from the geometric primitive described by the above parameter vector. In this situation fitting means to minimise the maximum distance of all points from the surface, a task involving numerical optimisation. This approach relies neither on the surface representation nor on the selection of geometric primitives used within this thesis, and can therefore be applied to other types of primitives and surfaces.

Fitting itself is performed after a “growth step”: for a given region of triangles an adjacent triangle is temporarily added to this current region. The triangle is accepted as belonging to the current region, if any vertex not already belonging to the region lies within a preset tolerance of the surface. Otherwise the surface parameters may not adequately represent the temporarily augmented region. In this case characteristic parameters need to be determined that minimise the

maximal distance of all points from the surface. The resulting minimisation problem is in general non-linear and non-smooth such that “classical” optimisation algorithms as the Levenberg-Marquard one are not applicable. Therefore in this thesis the minimisation problem has been tackled by a combined optimisation method encompassing an initial optimisation using a Genetic Algorithm and a subsequent Direct Search optimisation by Hooke and Jeeves. This approach employs the maximum distance as a measure of fit in contrast to least-squares methods used most often in previous work. Thus it allows a firm decision which points to add to and which ones to exclude from a specific region that grows on the surface during region growing.

Moreover it can be argued that this procedure has another advantage: when a point is tested for fitting to the characteristic parameters describing a primitive's surface it is expected to happen more often that it lies *a priori* within the preset tolerance (unless the point belongs to the border of a homogeneous region). The probability for this is expected to increase with the number of vertices in the region. This is due to the fact that the more points belong to a region the more precisely its characteristic parameters can be determined, and the more likely the data point under consideration is to lie within the “tolerance hull” of the geometric primitive. Such an approach may avoid a time-consuming refitting of characteristic parameters as it is in general required for a least-squares fit.

Furthermore the differences between successive and concurrent segmentation need to be discussed. Successive segmentation assigns triangles consecutively to regions so that a triangle belongs to only one region at a time. In this way triangles are assigned uniquely. On the other hand, concurrent segmentation may assign a surface triangle to several regions that are grown in parallel. Ambiguities that arise need to be resolved in a post-processing phase where, for example, the triangle is assigned to the largest region, or to the region where the triangle shows the best “goodness of fit”. The advantage of a concurrent segmentation strategy is that the decision of assigning triangles to regions is made after the region growing such that possible misassignments can be corrected. A good example of such a concurrent strategy is given in [Leonardis et al. 95] where only regions that grow in a pre-processing step sufficiently large are allowed to grow further followed by an overall selection of regions that form an “optimal” segmentation. The

drawback, however, consists in notably slower algorithmic performance. So in this thesis performance has been given priority to segmentation optimality.

#### 6.2.4 Segmentation results

Ideally an evaluative framework for surface segmentation – as intended with this project - is established as follows. Firstly, a 3D scene has to be composed of planes, spheres, cylinder, cones, and tori of specific dimensions such that all combinations of geometric primitives occur that allow for smooth joins (see Table 6.1). Such a scene is considered to be an appropriate framework for the comparison of segmenters that are able to extract parts of geometric primitives.

Next, this scene is scanned and the resulting point cloud converted to a triangulated surface. For short, this surface is referred to as primitive-synthesized surface. Thereafter, this surface needs to be segmented by an appropriate segmenter, and the resulting characteristic parameters determined for the extracted segments can then be compared to the initial parameter estimates. In particular, these results can be assessed in quality and quantity with respect to over-segmentation, under-segmentation, missed, and noise segments as defined in Section 2.2.1. Unfortunately, such a surface synthesis has not been possible for the present work.

Smooth join between	Plane	Sphere	Cylinder	Cone	Torus
Plane	–	–	$\delta_{\min} = \delta_{\max}$	$\delta_{\min} = \delta_{\max}$	$\delta_0$
Sphere	–	–	$\delta_{\min} = \delta_{\max}$	$\delta_{\min} = \delta_{\max}$	$\delta_{\min} = \delta_{\max}$
Cylinder	$\delta_0$	$\delta_{\max}$	$\delta_0$	$\delta_0$	$\delta_{\max}$
Cone	$\delta_0$	$\delta_{\max}$	$\delta_0$	$\delta_0$	$\delta_{\max}$
Torus	$\delta_0$	$\delta_{\min}$ and $\delta_{\max}$	$\delta_{\min}$ and $\delta_{\max}$	$\delta_{\max}$	$\delta_{\min}$ and $\delta_{\max}$

**Table 6.1:** *For each type of geometric primitive in the left column the direction of principal curvature is indicated along which it can be smoothly joined to one in the top row (“–” = non-existing smooth joins). Each of the geometric primitives apart from the plane is assumed to be convex (so that  $\kappa_{\max} > \kappa_{\min} > 0$  for the corresponding principal curvatures). The torus is assumed to have a larger major than minor radius.  $\delta_0$  denotes the direction of principal curvature that is associated with 0 principal curvature.*

Accordingly, the quality of the segmentation algorithm presented here can only be assessed on a higher level. So for the sample data sets presented in Section 4.6 neither noise segments nor under-segmentation occurred, and relatively few triangles have been missed. On the other hand some parts of geometric primitives are heavily over-segmented, especially those involving cones and cylinders on all data sets. This might be interpreted either as a failure of the parameter optimisation regarding the accuracy of the combined GA and Direct Search method, or as a too restrictive setting for the tolerance  $\tau$ . Another possible cause for the over-segmentation may be an inaccurately scanned surface: owing to varying angles of reflection, for example, a laser scanner could produce different measurement accuracies when scanning a plane and a cylinder with axis parallel to the ground, say. Most likely over-segmentation is caused by a combination of all three factors. Thus this adverse effect and in particular the scanning process require further investigation which is left to future work.

Another open problem concerns the misclassification of extracted segments. This may be because any cylinder, for example, is also a cone with very distant apex. A plane can be approximated locally to a sphere of very large radius. For these reasons the order in which instances of geometric primitives have been extracted has preferred “simple” shapes to “complex” ones.

Two interesting results not presented here in detail shall be summarised briefly. Firstly, the Direct Search optimisation method alone without employing a GA was found to lead to a significant over-segmentation because this method is not designed for escaping from local extrema. Secondly, the usage of a GA alone performs almost equally well in terms of the result as the combined approach with a small restriction in accuracy because of slower convergence near the optimal solution.

### **6.3 Fast extraction of planar segments**

The key idea of the fast plane extraction algorithm in Chapter 5 is to cut down the time-consuming plane parameter updates to a minimum. The algorithm itself is again based on region growing where adjacent triangles (i.e. triangles sharing an edge with the current region) are tested for inclusion in the region. As such triangles already share two vertices with the current planar region, only one vertex

needs to be tested by evaluating its distance from a plane that represents this region. This representative plane is given by a so-called *reference triangle*. If the vertex-to-plane distance test yields a distance exceeding a preset tolerance  $\tau$ , then this might indicate a tilt of the reference triangle. However, each time a vertex (and with it the corresponding triangle) is added to the current region  $\mathbf{R}$  an attempt is made to determine a triplet of vertices in  $\mathbf{R}$  spanning a triangle of largest possible area, here referred to as *expanded triangle*. Both reference triangle and expanded triangle are initialised by a seed triangle, which is also employed as the seed of the current planar region.

Now each time the above vertex-to-plane distance test exceeds  $\tau$ , the (possibly tilted) reference triangle may require replacement by a triangle representing the current region more accurately. So in this case the expanded triangle replaces the reference triangle.

The method described above is what has been called the *basic method* of planar extraction in this thesis. Because the basic method has some flaws, it necessitates further improvements. The first improvement concerns the distance of the vertices used for updating the expanded triangle. Some vertices may be in a large distance from the plane given by the reference triangle. This can lead to a triplet selection for the expanded triangle, which is not the best. Therefore only vertices within a given distance are used for the update of the expanded triangle. The distance chosen is the average distance of all the vertices from this plane multiplied by a constant somewhat larger than 1, such as 1.2, say.

The second improvement concerns the order of the surface triangles used as seed triangles. Because very often scan lines have imposed an order of the facets of a triangulated surface triangles near the boundary of a planar region are selected first as seed triangles. The vertices of such triangles may not represent a planar region very well. So it is suggested to select seed triangles rather more randomly, or in permuted order. In Section 5.2.3 a scheme has been presented providing some kind of “renumbering” of triangles in order to obtain better candidates for seed triangles. Another way might be to only select triangles as seeds if the direct neighbours (and possibly the ones of next higher order) are also of low curvature.

Extraction of planes from triangulated surface data employing the above method has proven to be reliable, accurate, fast, and robust even for noisy data though not perfect (e.g. consider the lower right facet of the left nut in Figure 5.9). Because this method merely exploits adjacency information, it is also suited to extracted planar regions from other data representations such as octrees, voxels, and polyhedral surfaces.

Moreover, taking the maximum distance of a vertex (or octree particle, or voxel, respectively) in a planar segment to the extracted plane allows an estimation of the measurement accuracy of the data. Because the tolerance  $\tau$  is determined automatically from this measure, over-segmentation and under-segmentation can be suppressed though not entirely prevented. For example, in combination with a lower threshold that a valid segment of a geometric primitive must have (`MIN_TRIANGLES`), a correctly determined tolerance can prevent the shell of a cylinder from being segmented into long thin strips of planes.

In cases where the data does not contain any planar regions at all this method is likely to fail. Then the measurement accuracy could be determined as the distance to another geometric primitive such as a cylinder, say. Fortunately, planar regions are present in the vast majority of surfaces designed for engineering purposes.

## **6.4 Implications and future work**

Overall, the work presented in this thesis represents a considerable step towards the aim of extracting shape information from surfaces so that engineers can use it for reverse engineering. The segmentation results in Sections 4.6 and 5.3 exhibit the capability of the presented approach to decompose a triangulated surface into individual segments of geometric primitives.

Therefore the next subsection explains the idea how to obtain a parametrisation from each extracted segment of a triangulated surface though it has not been one of the key issues of this work. Such a parametrisation can support the exchange of data between various software tools for surface design. The discussion then concludes with suggestions for potential improvements resulting in further work.

### **6.4.1 From a segmented to a parameterised surface**

The following brief sketch shall indicate the strategy how a segment-wise parameterisation might be achieved in principle. For this, each segment could be parameterised as explained in one of the following three cases.

1. If the segment consists of a single connected boundary curve, i.e. it has no “holes”, then at first divide the whole boundary into four connected curves of approximately the same length. Next interpolate, for example, the surface by a Bezier or spline surface.
2. If the segment consists of two separated boundary curves, i.e. it has a single “hole”, then “cut” the surface along a line from the exterior to the interior boundary. Denote the exterior boundary curve with A, the interior one with C, and let the line that connects A and C be referred to by B, the same line in opposite direction (the “return path”) by D. Then the area between A, B, C, and D may be interpolated in the same way as in the previous case. The possibly different lengths of the curves can be neglected.
3. If the segment consists of multiple separated boundary curves, i.e. it has multiple “holes”, then cut a path from the exterior boundary curve to one of the interior holes that is near to it. From there, successively cut a path to the next, until there remains a single hole that is cut only half way through (probably the most “interior” one). As above denote the exterior boundary curve by A, the most interior one by C, and the path from A to C by B, and the return path from C to A by D. Then apply one of the surface interpolation methods of the first case.

### **6.4.2 Potential improvements of the segmentation**

Despite the segmentation procedure presented in this thesis, the segmentation problem will still offer opportunities for improvements. Amongst these the following topics outline the ideas to tackle in future research.

Firstly, various methods of curvature estimation appropriate for triangulated surface data need to be compared to each other with respect to accuracy and

performance. This concerns in particular methods that interpolate the triangle vertices with continuous surface patches as well as methods that estimate curvature discretely.

Further estimates of characteristic parameters of geometric primitives on triangulated surfaces or lower level data representations such as point data need to be developed and investigated. Of particular interest is their robustness to noise in the data. Additionally, it needs to be investigated how various parameter estimation methods perform in comparison on a complete scene of geometric primitives with smooth joins such as presented in Table 6.1.

A crucial part of the present segmentation remains the optimisation of the characteristic parameters, here formulated as a minimax problem. Rather than the combined GA and Direct Search optimisation the barrier function method of [Polak et al. 92] may be an appropriate, more efficient alternative. Further investigation is also required for the problem to determine how the order of extracting geometric primitives affects the segmentation result. In particular it needs to be examined whether there exists an optimal extraction order for an arbitrary input triangulated surface.

Apart from the successive segmentation strategies used for this project, concurrent strategies should be implemented (such as the recover-and-select paradigm of [Leonardis et al. 95]) and their output compared to each other. For example, a somewhat milder version of the recover-and-select paradigm may consist of minimising the expression

$$n_{\tau} (1 + w \tau)$$

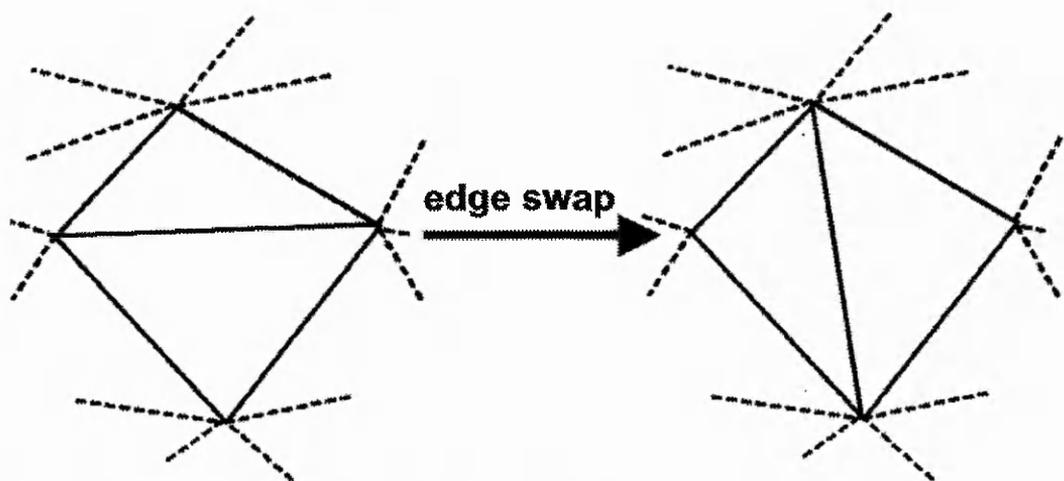
where  $n_{\tau}$  denotes the number of segments obtained from a segmenter,  $\tau \geq 0$  the maximum distance of a point to the underlying part of a geometric primitive, and  $w > 0$  a preset weight.

Finally, a further improvement concerns the joins between identified segments as they often appear to be “smoothed” so that the segmenter leaves gaps between them resulting in an incomplete segmentation. This occurs, for example, when data has been pre-processed (e.g. filtered), or sampled with too small density, which means that no data points have been acquired precisely on surface edges or

creases. Hence it remains an additional task to restore the original potentially “sharp joins” between adjacent segments. A similar task concerns the accurate determination of intersection curves at locations where two adjacent segments meet. In order to cope with these problems, after the segmentation task as described here a post-processing phase may be necessary. In this phase data points along the non-smooth joins of adjacent segments could be added, and the corresponding parts of a triangulated surface could be re-triangulated to improve its overall structure. Afterwards, the resulting triangles could be assigned to the already identified segments so that all gaps between adjacent segments were closed and the “sharp joins” restored.

### 6.4.3 The need for re-triangulation of a triangulated surface

An outcome of this project is that in some cases the segmentation result may require an improvement with respect to triangles near the boundaries of adjacent extracted segments. In fact, some regions grown on a triangulated surface could have been enlarged if the triangulation had been “improved” by edge swap operations (see Figure 6.4.3). In this context, the Author has initiated and conducted research efforts to improve the quality of a triangulation for subsequent extraction of geometric primitives (see [Lakin 00]).



**Figure 6.4.3:** *The “edge swap” operation on a pair of triangles*

In the following the results are summarised briefly: the generation of a triangulated surface from an unstructured point cloud (i.e. scanned surface data) is a fairly complicated task requiring further improvements of POMOS, i.e. the CAD/CAM tool that has been used for this project. It is recalled from Section

1.2.3 that the “surface generation by triangulation” method currently implemented in POMOS processes 2½D data only. The surface triangles are created by connecting data points by edges that have been acquired in parallel lines called scan-lines using a simple angle minimising criterion. The Author has developed criteria for a pair of adjacent triangles to “swap” their edges in order to improve the quality of a triangulated surface by minimising, for example, the total length of edges of surface triangles. Further criteria involve the minimisation of the length of the longest edge of two adjacent triangles or, similarly, the maximisation of their shortest edges. Another efficient method is to swap an edge of a pair of adjacent triangles so that the midpoint of the swapped edge is more distant from the viewpoint than the midpoint of the original “central” edge.

The above measures are valuable suggestions for improving a triangulated surface in a way that allows larger segments to be extracted. Algorithms for the practical realisation of all the above tasks have been developed, validated, and their practicability has been demonstrated.

## 7 Conclusions

The segmentation of a triangulated surface into parts of geometric primitives in the presence of measurement inaccuracies is a delicate task. As described in Section 1.6, the aim of this project has been to recover shape information from such a surface in terms of parts from a given set of geometric primitives. In many CAD/CAM applications a set encompassing the primitives plane, sphere, cylinder, cone, and torus has emerged as desirable for designers and engineers, and was thus chosen for the present project. Each of these primitives is uniquely determined by its characteristic parameters, and consequently the extraction of these parameters is one of the tasks associated with the recovery of shape information.

In order to model complex shapes easily and quick by CAD/CAM the creation of a digital representation from a physical object compares favourably to pure computer generated modelling. For the creation of such a representation a process of reverse engineering needs to be performed, whereby the object is scanned - usually from multiple viewpoints - by a laser scanner or a similar device. The resulting 2½D range image from each scan then comprises a large set of data points representing the surface of the object. However, every measurement produces errors, so the data points are subject to a small deviation in 3D space from the virtual surface. Hence each range image must be regarded as a discrete surface approximation. Because each of these images relates to a specific arrangement of scanner and object, scans from different perspectives result in images of different surface parts. Multiple range images require an image registration, i.e. a process of aligning the sets of points in a common coordinate frame, where a number of 2½D data sets may be merged to form a 3D data set. In all cases during reverse engineering a surface needs to be generated from the data at some stage, either before or after the registration. Triangulated surfaces are often used because there are several advantages associated with this representation. For the manipulation of a triangulated surface a high-level description is very useful so that complete surface parts can be modified rather than individual triangles. Such a high-level description can be obtained by segmenting the triangulated surface into parts of geometric primitives as stated above. This involves the extraction of the characteristic parameters for each segment and the determination of its corresponding boundary curve(s). The work presented in this

thesis represents a significant contribution towards the aim of segmenting a triangulated surface in just this way.

Suppose that a triangulated surface is given with information associated with every triangle about normals and “neighbours”, i.e. triangles that are directly connected to it by an edge. Algorithms have been developed by the Author for segmentation of such a surface into parts of geometric primitives effecting the following two steps:

1. curvature estimation, and
2. extraction of parts of planes, spheres, cylinders, cones, and tori.

Firstly, curvature (a concept of differential geometry) is estimated to gain initial shape information about the triangulated surface ([Sacchi et al. 99]). Secondly, parts of the above geometric primitives are extracted by using a “region growing” approach, which has been assessed as the most appropriate for the given task. Region growing encompasses three phases: seed region selection, surface fitting and segment labelling. Initially a seed region is selected and then it is allowed to grow, i.e. new points nearby are added to the current region if they satisfy a specific shape hypothesis. During the growing the characteristic parameters of each geometric primitive are adjusted, so that a specific surface shape is fitted to a data subset. This phase is usually referred to as “surface fitting”. When a region cannot grow any further, it needs to be validated and to be labelled as a segment of a specific shape to prevent interpretation as a different shape.

Region growing in the context of surface fitting can be summarised as follows. The type of geometric primitive for region growing is determined by an initial shape hypothesis, which consists of the selection of specific types of geometric primitives and the determination of its characteristic parameters. The latter task can be accomplished by exploitation of the curvature estimates. Unlike in [Besl & Jain 88] the present approach is not based on quadric modelling functions, and for this reason the type of geometric primitive is not allowed to change during region growing. Instead parts of geometric primitives are extracted in order of their complexity, which is expressed by the number of their characteristic parameters required to determine a unique instance of each of them (see first paragraph of this chapter). During region growing, i.e. when a specific surface shape is fitted to the data, the characteristic

parameters of each primitive are likely to need adjustment (for example because of quantisation effects and measurement inaccuracies in the data points). This parameter adjustment involves numerical optimisation of an implicit function modelling each type of geometric primitive. When no further point can be added to the current region without violating the shape hypothesis, growing of the current region stops. Then the accrued region is separated from the data, and an attempt is made to grow a new region for the remaining set of points. Region growing for the selected type of geometric primitive terminates after all data points (individually or in groups) have been employed as initial seed regions. The overall region growing terminates after all types have been used for the generation of a shape hypothesis.

Original contributions to research have been made by the Author as follows.

Previously curvature computations have most often been applied to smooth surfaces. Therefore a novel method has been established for estimating curvature of triangulated surfaces. The method performs well compared to the few other methods of curvature estimation for polyhedral surfaces. Curvature on surfaces is given a sign in order to distinguish between convex and concave ones. Examinations by the Author have shown that the formula for the determination of the sign of curvature in [Flynn & Jain 89] sometimes gives wrong results. This thesis has also presented a correction of this error-prone formula.

Based on these curvature estimates, a novel method for the calculation of initial characteristic parameters for the geometric primitives under consideration has been developed by the Author. These calculations do not require the solution of a system of linear equations such as proposed in [Lukács et al. 98]. Segments for all types of geometric primitives have been successfully extracted from triangulated surface data by combining a genetic algorithm and a direct search algorithm for the surface fitting, which involves numerical optimisation of a minimax problem. By comparing the type of an extracted geometric primitive to the one expected in a particular surface area, the correspondence is fairly high.

For fast extraction of planes a special method, which requires merely direct edge-connectivity information, has been developed by the Author and has been shown to

be efficient ([Sacchi et al. 00]). This method can even be applied to data other than triangulated surfaces.

Further work may be spent on the investigation of a recover-and-select paradigm ([Leonardis et al. 95]), which involves concurrent region growing for different types of geometric primitives, when applied to the present approach. Although the minimax optimisation problem for the surface fitting phase has been solved in principle, further research is necessary to find a more efficient optimisation technique. This can improve the overall performance of the proposed segmentation algorithm. Moreover, the problem of automated optimisation of triangulated meshes for improved extraction of parts of geometric primitives has just been tackled, but still requires further investigation.

## 8 References

- [Ashbrook et al. 97] Ashbrook AP, Fisher RB, Robertson C, Werghi N, *Segmentation of Range Data into Rigid Subsets using Planar Surface Patches*, Proceedings on the British Machine Vision Conference, Essex, Sep 97, pp. 530-539
- [Ballard & Brown 82] Ballard DH, Brown CM, *Computer Vision*, Englewood Cliffs, New Jersey, Prentice Hall, 1982
- [Besl & Jain 88] Besl PJ, Jain RC, *Segmentation Through Variable-Order Surface Fitting*, IEEE Transactions on Pattern Analysis & Machine Intelligence, 1988, Vol. 10, No. 2, pp. 167-192
- [Besl & McKay 92] Besl PJ, McKay DN, *A Method for Registration of 3D Shapes*, IEEE Transactions on Pattern Analysis & Machine Intelligence, 1992, Vol. 14, No. 2, pp. 239-256
- [Biswas et al. 95] Biswas PKR, Biswas SS, Chatterji BN, *An SIMD Algorithm for Range Image Segmentation*, Pattern Recognition, 1995, Vol. 28, No. 2, pp. 255-267
- [Bolle & Sabbah 87] Bolle RM, Sabbah D, *Differential geometry applied to least-square error surface approximation*, Proceedings on SPIE Optical and Digital Pattern Recognition, Los Angeles, 1987, Vol. 754, pp. 117-127
- [Brehm & Kühnel 81] Brehm U, Kühnel W, *Smooth Approximation of Polyhedral Surfaces with Respect to Curvature Measures*, in Ferus et al. (Eds.), *Global Differential Geometry and Global Analysis*, Lecture Notes in Mathematics, Vol. 838, Springer publishers, 1981
- [Bronstein & Semendjajew 85] Bronstein IN, Semendjajew KA, *Taschenbuch der Mathematik*, 22nd edition, Frankfurt/Main, Thun, 1985
- [Campbell & Flynn 98] Campbell R, Flynn PJ, *Model and Range Image Features for Free-Form Object Recognition*, Proceedings of the Vision Interface, Vancouver, Jun 17-20 1998, pp. 173-180
- [Certain & Stuetzle 99] Certain A, Stuetzle W, *Automatic Body Measurement for Mass Customization of Garments*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 405-412

- [Charalambous & Conn 78] Charalambous C, Conn AR, *An Efficient Method to Solve the Minimax Problem Directly*, SIAM J. of Numerical Analysis, 1978, Vol. 15, No. 1, pp. 162-187
- [Chen & Liu 97] Chen YH, Liu CY, *Robust Segmentation of CMM Data Based on NURBS*, International Journal of Advanced Manufacturing Technology, 1997, Vol. 13, pp 530-534
- [Chen & Liu 99] Chen YH, Liu CY, *Quadric Surface Extraction using Genetic Algorithms*, Computer-Aided Design, 1999, Vol. 31, No. 2, pp 101-110
- [Choi 91] Choi BK, *Surface Modeling for CAD/CAM*, Elsevier, 1991
- [Dafas et al. 00] Dafas P, Kompatsiaris I, Strintzis MG, *Optimal Hierarchical Adaptive Mesh Construction Using FCO Sampling*, Proceedings of the IEEE Conference on Information Visualization, London, July 19-21 2000, pp. 331-336
- [Delorme et al. 99] Delorme S, Petit Y, de Guise JA, Aubin CÉ, Labelle H, Landry C, Dansereau J, *Three-Dimensional Modelling and Rendering of the Human Skeletal Trunk from 2D Radiographic Images*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 497-505
- [Faugeras et al. 83] Faugeras OD, Hebert M, Pauchon E, *Segmentation of Range Data into Planar and Quadratic Patches*, Proceedings of the 3<sup>rd</sup> Conference on Computer Vision and Pattern Recognition, Arlington, VA, 1983, pp. 8-13
- [Fischer 99] Fischer D, *Rekonstruktion dreidimensionaler Oberflächenmodelle aus Sequenzen segmenturverrauschter Tiefenbilder*, Düsseldorf, VDI-Verlag, 1999 (Fortschritt-Berichte / VDI: Reihe 10, Informatik, Kommunikationstechnik; Nr. 606)
- [Fisher et al. 93] Fisher RB, Fitzgibbon A, Waite, Orr ML, Trucco E, *Recognition of Complex 3-D Objects from Range Data*, Progress in Image Analysis and Image Processing (Proceedings of the 7<sup>th</sup> International Conference on Image Analysis and Processing), Impedovo S (Ed.), Bari (Italy), Sept 1993, pp 509-606

- [Fisher et al. 97] Fisher RB, Fitzgibbon AW, Eggert D, *Extracting Surface Patches from Complete Range Descriptions*, Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling, Ottawa, May 1997, pp. 148-155
- [Fisher et al. 99] Fisher RB, Ashbrook AP, Robertson C, Werghi N, *A Low-Cost Range Finder using a Visually Located, Structured Light Source*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 24-33
- [Fletcher 87] Fletcher R, *Practical Methods of Optimization*, J. Wiley & Sons, Chichester, 1987
- [Flynn & Jain 89] Flynn PJ, Jain AK, *On Reliable Curvature Estimation*, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Rosemont, Illinois (USA), Jun 6-9 1989, pp. 110-116
- [Fusiello et al. 99] Fusiello A, Giannitrapani R, Isaia V, Murino V, *Virtual Environment Modeling by Integrated Optical and Acoustic Sensing*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 437-446
- [Gill et al. 81] Gill PE, Murray W, Wright MH, *Practical Optimization*, Academic Press, London 1981
- [Gonzalez & Woods 92] Gonzalez RC, Woods RE, *Digital Image Processing*, Reading, Massachusetts, Addison-Wesley, 1992
- [Häfele & Hellmann 96] Häfele KH, Hellmann M, *3D-Meßdatenaufbereitung und Weiterverarbeitung*, 2<sup>nd</sup> Workshop Optische 3D-Formerfassung großer Objekte, Technical Academy Esslingen (Germany), 25.-26.1.1996
- [Häfele 96] Häfele KH, *POMOS – POint based MOdelling System*, Contribution to *Second Workshop On Current CAX-Problems*, Tauberbischofsheim/Germany, 26<sup>th</sup>-29<sup>th</sup> February 1996; also published in: Hoschek, J, Dankwort, W (Eds.), *Reverse Engineering*, Verlag B.G. Teubner, Stuttgart/Germany, 1996
- [Hald & Madsen 81] Hald J, Madsen K, *Combined LP and Quasi-Newton Methods for Minimax Optimization*, Mathematical Programming Study, North-Holland Publishing Company, 1981, Vol. 20, pp. 49-62

- [Hamann 93] Hamann B, *Curvature Approximation for Triangulated Surfaces*, Computing Supplementum, 1993, Vol. 8, pp. 139-153
- [Han et al. 87] Han J, Volz RA, Mudge TN, *Range Image Segmentation and Surface Parameter Extraction for 3D Object Recognition of Industrial Parts*, Proceedings of the IEEE International Conference on Robotics and Automation, 1987, Vol. 1, pp. 380-386
- [Hebert & Ponce 82] Hebert M, Ponce J, *A New Method for Segmenting 3-D Scenes into Primitives*, Proceedings of the International Conference on Pattern Recognition, Munich, 1982, pp. 836-838
- [Hilton et al. 95] Hilton A, Illingworth J, Windeatt T, *Statistics of Surface Curvature Estimates*, Pattern Recognition, 1995, Vol. 28. No. 8, pp. 1201-1221
- [Hoffman & Jain 87] Hoffman RL, Jain AK, *Segmentation and Classification of Range Images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1987, Vol. 9, No. 5, pp. 608-620
- [Holland 75] Holland JH, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Michigan, 1975
- [Hooke & Jeeves 61] Hooke R, Jeeves TA, *Direct search solution for numerical and statistical problems*, Journal of the Association for Computing Machinery, 1961, Vol. 8, pp. 212 – 229
- [Hoover et al. 96] Hoover A, Jeanbaptiste G & others, *An Experimental Comparison of Range Image Segmentation Algorithms*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1996, Vol. 18, No. 7, pp. 673-689
- [Hoppe 94] Hoppe H, *Surface Reconstruction from Unorganized Points*, PhD thesis, University of Washington, 1994
- [Horowitz & Pavlidis 74] Horowitz SL, Pavlidis T, *Picture Segmentation by a Directed Split-and-Merge Procedure*, Proceedings of the 2<sup>nd</sup> International Joint Conference on Pattern Recognition, 1974, pp. 424-433
- [Hoschek & Lasser 93] Hoschek J, Lasser D, *Fundamentals of Computer Aided Geometric Design*, AK Peters, Wellesley, Massachusetts, 1993

- [Hoschek 96] Hoschek J, *Benchmarks*, in: Hoschek, J, Dankwort, W (Eds.), *Reverse Engineering*, Verlag B.G. Teubner, Stuttgart/Germany, 1996, pp. 160-163
- [Hoschek et al. 98] Hoschek J, Dietz U, Wilke W, *A Geometric Concept of Reverse Engineering of Shape: Approximation and Feature Lines*, published in: Daehlen M, Lyche T, Schumaker LL (Eds.), *Mathematical Methods for Curves and Surfaces II*, 1998, pp. 253-262
- [Jiang & Bunke 94] Jiang XY, Bunke H, *Fast Segmentation of Range Images into Planar Regions by Scan Line Grouping*, Machine Vision and Applications, 1994, Vol. 7, No. 2, pp 115-122
- [Ke et al. 97] Ke Q, Jiang T, De Ma S, *A Tabu Search Method for Geometric Primitive Extraction*, Pattern Recognition Letters, 1997, Vol. 18, No. 14, pp. 1443-1451
- [Kehtarnavaz 88] Kehtarnavaz N, RJP de, *A 3-D Contour Segmentation Scheme Based on Curvature and Torsion*, IEEE Transactions on Pattern Analysis & Machine Intelligence, 1988, Vol. 10, No. 2, pp. 707-713
- [Krsek et al. 98] Krsek P, Lukács G, Martin RR, *Algorithms for Computing Curvatures from Range Data*, published in: The Mathematics of Surfaces, Cripps R (ed.), 1998, pp. 1-16
- [Lakin 00] Lakin P, *Triangle Refinement in POMOS*, Final Report at the FZK, July 2000
- [Lee 00] Lee CK, *Automatic Metric Advancing Front Triangulation over Curved Surfaces*, Engineering Computations, 2000, Vol. 17, No. 1, pp. 48-74
- [Leonardis et al. 95] Leonardis A, Gupta A, Bajcsy R, *Segmentation of Range Images as the Search for Geometric Parametric Models*, International Journal of Computer Vision, 1995, Vol. 14, pp. 253-277
- [Levoy 99] Levoy M, *The Digital Michelangelo Project*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 2-11
- [Lorenz & Krahnstöver 99] Lorenz C, Krahnstöver N, *3D Statistical Shape Models for Medical Image Segmentation*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital

Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 414-423

- [Lukács et al. 98] Lukács G, Marshall AD, Martin RR, *Faithful Least-Squares Fitting of Spheres, Cylinders, Cones and Tori for Reliable Segmentation*, Lecture Notes in Computer Science, Springer publishers, 1998, Vol. 1406, pp. 671-686
- [Maître et al. 90] Maître G, Hügli H, Tièche F, Amann JP, *Range Image segmentation based on function approximation*, Close-Range Photogrammetry Meets Machine Vision, Ed. A Gruen & E Baltsavias (1990), 1990, SPIE Vol. 1395, pp. 275-282
- [Mirza 95] Mirza MJ, *Robust Sequential Surface Based Object Segmentation in Noisy Range Data*, Proceedings of the 5<sup>th</sup> International Conference on Image Processing and its Application, July 4-6 1995, Heriot-Watt University, Edinburgh (UK), pp. 271-275
- [Ng et al. 95] Ng I, Illingworth J, Jones G, *A Novel Method for Segmentation of Cones and Cylinders from Geometrically Fused Depth Maps*, Proceedings of the 5<sup>th</sup> International Conference on Image Processing and its Applications, Edinburgh, Jul 4-6 1995, pp. 544-548
- [Nguyen et al. 99] Nguyen V-D, Nzomigni V, Stewart CV, *Fast and Robust Segmentation of 3D Surfaces Using Low Curvature Patches*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 201-208
- [Polak et al. 92] Polak E, Higgins JE, Mayne DQ, *A Barrier Function Method for Minimax Problems*, Mathematical Programming, North Holland, 1992, Vol. 54, pp. 155-176
- [Powell 98] Powell MJD, *Direct search algorithms for optimization calculations*, to be published in: Acta Numerica, Cambridge University Press, 1998, Vol. 7
- [Powell et al. 98] Powell MW, Bowyer K, Jiang X, Bunke H, *Comparing Curved-Surface Range Image Segmenters*, Proceedings of the 6<sup>th</sup> International Conference on Computer Vision, Bombay, 1998, pp. 286-291
- [Pulli 99] Pulli K, *Multiview Registration for Large Data Sets*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999,

pp. 160-168

- [Robinette et al. 99] Robinette KM, Daanen H, Paquet E, *The Caesar Project: A 3-D Surface Anthropometry Survey*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 380-386
- [Roth 99] Roth G, *Registering two Overlapping Range Images*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 191-200
- [Roth & Levine 90] Roth G, Levine MD, *Segmentation of Geometric Signals using Robust Fitting*, Proceedings of the 10<sup>th</sup> International Conference on Pattern Recognition, Atlantic City, Jun 16-21 1990, pp. 826-831
- [Roth & Levine 91] Roth G, Levine MD, *A Genetic Algorithm for Primitive Extraction*, Proceedings of the 4<sup>th</sup> International Conference on Genetic Algorithms, San Diego, California, 1991, pp. 487-494
- [Roth & Levine 93] Roth G, Levine MD, *Extracting Geometric Primitives*, Computer Vision, Graphics & Image Processing - Image Understanding, 1993, Vol. 58, No 1, pp 1-22
- [Sacchi et al. 99] Sacchi R, Poliakoff JF, Thomas PD, *Curvature Estimation for Segmentation of Triangulated Surfaces*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 536-543
- [Sacchi et al. 00] Sacchi R, Poliakoff JF, Thomas PD, Häfele KH, *Improved Extraction of Planar Segments for Scanned Surfaces*, Proceedings of the IEEE Conference on Information Visualization, London, July 19-21 2000, pp. 325-330
- [Shihong et al. 99] Shihong L, Sumi Y, Kawade M, Tomita F, *Building Facial Models and Detecting Face Pose in 3D Space*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 398-404
- [Stoer 83] Stoer J, *Einfuehrung in die Numerische Mathematik I (Introduction into Numerical Mathematics I)*, 1983, Springer publishers New York Heidelberg Berlin Tokyo, 4th edition

- [Tanaka et al. 98] Tanaka HT, Ikeda M, Chiaki H, *Curvature-based Face Surface Recognition Using Spherical Correlation – Principal Directions for Curved Object Recognition*, Proceedings of the 3<sup>rd</sup> IEEE International Conference on Automatic Face and Gesture Recognition, Nara (Japan), Apr 14-16 1998, pp. 372-377.
- [Taubin 91] Taubin G, *Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1991, Vol. 13, No. 11, pp. 1115-1138
- [Taylor et al. 89] Taylor RW, Savini M, Reeves AP, *Fast Segmentation of Range Imagery into Planar Regions*, Computer Vision, Graphics & Image Processing, 1989, Vol. 45, No. 1, pp. 42-60
- [Trucco & Fisher 95] Trucco E, Fisher RB, *Experiments in Curvature-Based Segmentation of Range Data*, IEEE Transactions on Pattern Analysis & Machine Intelligence, 1995, Vol. 17, No. 2, pp. 177-182
- [Várady et al. 97] Várady T, Martin RR, Cox J, *Reverse Engineering of geometric models - an introduction*, Computer-Aided Design, 1997, Vol. 29 No. 4, pp. 255-268
- [Veelaert 97] Veelaert P, *Constructive Fitting and Extraction of Geometric Primitives*, Graphical Models & Image Processing, 1997, Vol. 59, No. 4, pp. 233-251
- [Werghe et al. 99] Werghe N, Fisher RB, Ashbrook AP, Robertson C, *Faithful Recovering of Quadric Surfaces from 3D Range Data*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 280-289
- [Wilke 94] Wilke W, *Segment-Trennungen bei der Approximation von Kurven und Flaechen mit B-splines (Segment Separations for the Approximation of Curves and Surfaces using B-splines)*, Diploma thesis, TH Darmstadt (Germany), 1994
- [Wolfart et al. 99] Wolfart E, Sequeira V, Ng K, Butterfield S, Gonçalves JGM, Hogg D, *Hybrid Approach to the Construction of Triangulated 3D Models of Building Interiors*, Proceedings of the 1<sup>st</sup> International Conference on Computer Vision Systems, Las Palmas, Gran Canaria (Spain), Jan 13-15 1999, published as "Computer

Vision Systems", Springer publishers, 1999, Vol. 1542, pp. 489-508

- [Wright et al. 96] Wright M, Fitzgibbon A, Giblin PJ, Fisher RB, *Convex Hulls, Occluding Contours, Aspect Graphs and the Hough Transform*, Image and Vision Computing, 1996, Vol. 14, pp. 627-634
- [Yemez & Schmitt 99] Yemez Y, Schmitt F, *Progressive Multilevel Meshes from Octree Particles*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modeling, Ottawa, Oct 4-8 1999, pp. 290-299
- [Yokoya & Levine 91] Yokoya N, Levine MD, *A Hybrid Approach to Range Image Segmentation Based on Differential Geometry*, Journal of Information Processing, 1991, Vol. 14, No. 4, pp. 516-524
- [Yu et al. 94] Yu X, Bui TD, Krzyżak A, *Robust Estimation for Range Image Segmentation and Reconstruction*, IEEE Journal of Transactions on Pattern Analysis & Machine Intelligence, 1994, Vol. 16, No. 5, pp. 530-538
- [Zhao & Zhang 97] Zhao D, Zhang X, *Range-Data-Based Object Surface Segmentation via Edges and Critical Points*, IEEE Transactions on Image Processing, 1997, Vol. 6, No. 6, pp.826-830
- [Zucker 76] Zucker SW, *Region Growing: Childhood and Adolescence*, Computer Graphics and Image Processing, 1976, Vol. 5, pp. 382-399

## Appendix

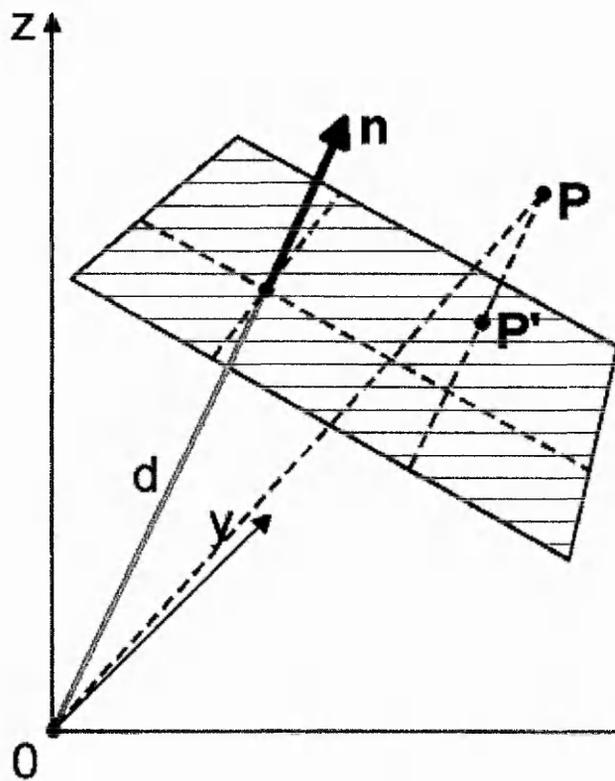
### A Modelling point-surface distances for each of the geometric primitives relevant to this project

For each of the functions  $f_{\text{pl}}(\mathbf{P}; \mathbf{X}_{\text{pl}})$ ,  $f_{\text{sph}}(\mathbf{P}; \mathbf{X}_{\text{sph}})$ ,  $f_{\text{cyl}}(\mathbf{P}; \mathbf{X}_{\text{cyl}})$ ,  $f_{\text{con}}(\mathbf{P}; \mathbf{X}_{\text{con}})$ , and  $f_{\text{tor}}(\mathbf{P}; \mathbf{X}_{\text{tor}})$  that are used in Section 2.6.1, Equation (2.14), to model the distance from a point  $\mathbf{P}$  to the surface of each of the geometric primitives in  $\mathbf{G} = \{\text{pl, sph, cyl, con, tor}\}$  a derivation is presented here. In each case the function determines the distance from  $\mathbf{P}$  to the nearest point  $\mathbf{P}'$  on the surface. So each of the above functions can be expressed by

$$f_{\mathbf{g}}(\mathbf{P}; \mathbf{X}_{\mathbf{g}}) = \|\mathbf{P} - \mathbf{P}'\|, \quad (\text{A.1})$$

where  $\mathbf{P}'$  is on the surface of  $\mathbf{g}$  such that it minimises the distance to  $\mathbf{P}$ , and  $\mathbf{X}_{\mathbf{g}}$  denotes the vector of characteristic parameters of  $\mathbf{g}$ . The aim of the subsequent calculations is to express the point  $\mathbf{P}'$  in terms of  $\mathbf{P}$  and the characteristic parameters in  $\mathbf{X}_{\mathbf{g}}$ . With this, each of the above functions can be determined instantaneously by virtue of Equation (A.1).

## A.1 The distance of a point to a plane



**Figure A.1:** *Determining the distance from a point  $P$  to the nearest point  $P'$  on a plane*

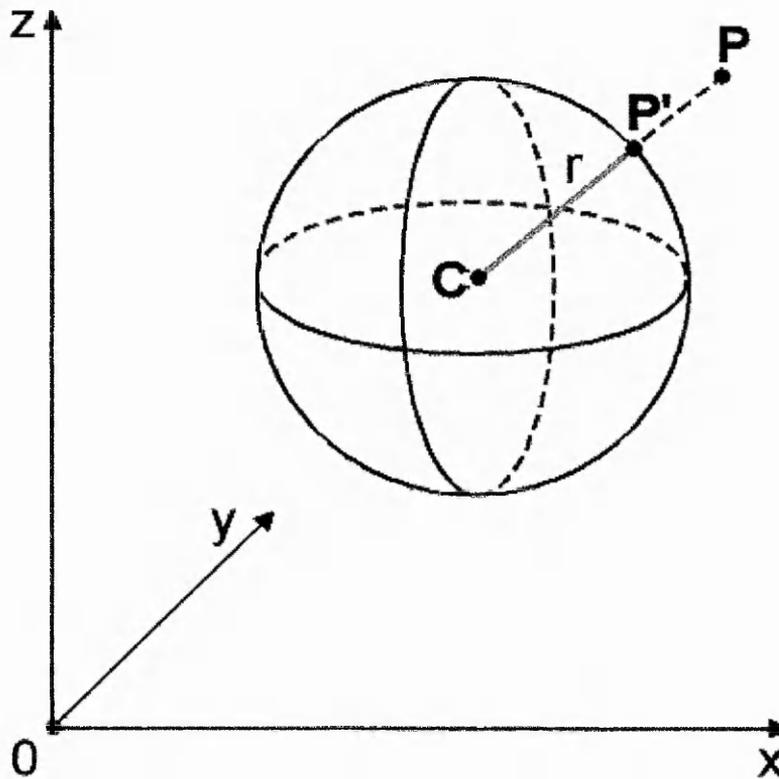
The characteristic parameters of the plane are a surface normal  $\mathbf{n}$  and the nearest distance  $d$  to the origin. By examining Figure A.1 the point  $P'$  as the nearest point to  $P$  on the plane can be determined by

$$\mathbf{P}' = d \mathbf{n} + \mathbf{P} - \langle \mathbf{P}, \mathbf{n} \rangle \mathbf{n},$$

and thus

$$\begin{aligned} \|\mathbf{P} - \mathbf{P}'\| &= \|d \mathbf{n} - \langle \mathbf{P}, \mathbf{n} \rangle \mathbf{n}\| \\ &= |d - \langle \mathbf{P}, \mathbf{n} \rangle|. \end{aligned}$$

## A.2 The distance of a point to a sphere



**Figure A.2:** *Determining the distance from a point  $P$  to the nearest point  $P'$  on a sphere*

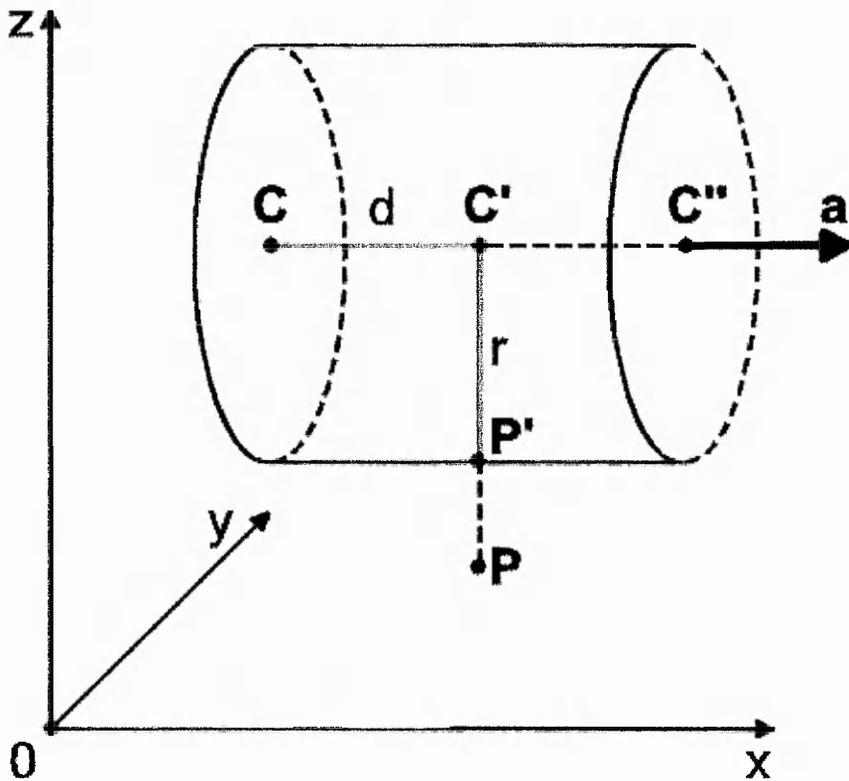
The characteristic parameters of a sphere include its centre  $C$  and its radius  $r$ . By examining Figure A.2 the point  $P'$  as the nearest point to  $P$  on the sphere can be calculated by

$$P' = C + r(P - C) / \|P - C\|$$

and thus

$$\begin{aligned} \|P - P'\| &= \|P - C - r(P - C) / \|P - C\|\| \\ &= \|(P - C)(1 - r / \|P - C\|)\| \\ &= \|P - C\| |1 - r / \|P - C\|| \\ &= |\|P - C\| - r|. \end{aligned}$$

### A.3 The distance of a point to a cylinder



**Figure A.3:** *Determining the distance from a point  $P$  to the nearest point  $P'$  on a cylinder*

The characteristic parameters of a cylinder are given by a unit vector  $\mathbf{a}$  in the direction of its axis, a point  $C$  on its axis, and its radius  $r$ . By examining Figure A.3 the point  $P'$  as the nearest point to  $P$  on the cylinder can be calculated as follows:

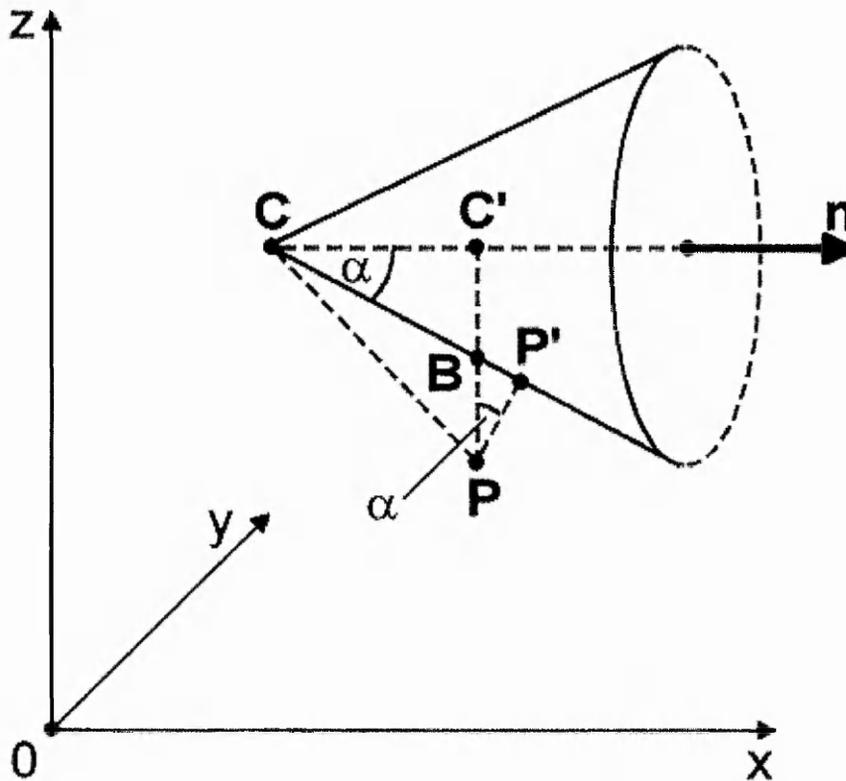
$$\mathbf{C}' = \mathbf{C} + \langle \mathbf{P} - \mathbf{C}, \mathbf{a} \rangle \mathbf{a},$$

$$\mathbf{P}' = \mathbf{C}' + r(\mathbf{P} - \mathbf{C}') / \|\mathbf{P} - \mathbf{C}'\|,$$

and thus

$$\begin{aligned} \|\mathbf{P} - \mathbf{P}'\| &= \|\mathbf{P} - \mathbf{C}' - r(\mathbf{P} - \mathbf{C}') / \|\mathbf{P} - \mathbf{C}'\|\| \\ &= \|\mathbf{P} - \mathbf{C}'\| \left| 1 - r / \|\mathbf{P} - \mathbf{C}'\| \right| \\ &= \|\mathbf{P} - \mathbf{C}'\| \left| 1 - r / \|\mathbf{P} - \mathbf{C}'\| \right| \\ &= \left| \|\mathbf{P} - \mathbf{C}'\| - r \right| \\ &= \left| \|\mathbf{P} - \mathbf{C} - \langle \mathbf{P} - \mathbf{C}, \mathbf{a} \rangle \mathbf{a}\| - r \right|. \end{aligned}$$

#### A.4 The distance of a point to a cone



**Figure A.4:** Determining the distance from a point  $P$  to the nearest point  $P'$  on a cone

The characteristic parameters of a cone encompass a unit vector  $\mathbf{a}$  in the direction of its axis, the apex  $C$  on its axis, and an opening angle  $2\alpha$ . As is illustrated in Figure A.4 the point  $P'$  as the nearest point to  $P$  on the cone can be obtained from

$$\mathbf{C}' = C + \langle \mathbf{P} - C, \mathbf{a} \rangle \mathbf{a},$$

$$\cos \alpha = \|\mathbf{C}' - C\| / \|\mathbf{B} - C\| = \|\mathbf{P} - \mathbf{P}'\| / \|\mathbf{P} - \mathbf{B}\|,$$

and thus

$$\|\mathbf{B} - C\| = \|\mathbf{C}' - C\| / \cos \alpha$$

and

$$\|\mathbf{P} - \mathbf{P}'\| = \|\mathbf{P} - \mathbf{B}\| \cos \alpha,$$

$$\|\mathbf{P} - \mathbf{B}\| = \|\mathbf{P} - \mathbf{C}'\| - \|\mathbf{B} - \mathbf{C}'\|$$

$$= (\|\mathbf{P} - C\|^2 - \|\mathbf{C}' - C\|^2)^{1/2} - (\|\mathbf{B} - C\|^2 - \|\mathbf{C}' - C\|^2)^{1/2}$$

$$= (\|\mathbf{P} - C\|^2 - \|\mathbf{C}' - C\|^2)^{1/2} - (\|\mathbf{C}' - C\|^2 (1/\cos^2 \alpha - 1))^{1/2}$$



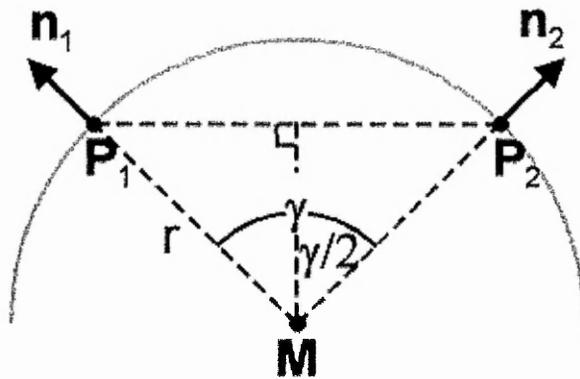
$$\begin{aligned}
&= \left\| (\mathbf{P} - \mathbf{M}) \left( 1 - r / \|\mathbf{P} - \mathbf{M}\| \right) \right\| \\
&= \|\mathbf{P} - \mathbf{M}\| \left| 1 - r / \|\mathbf{P} - \mathbf{M}\| \right| \\
&= \left| \|\mathbf{P} - \mathbf{M}\| - r \right| \\
&= \left| \left\| \mathbf{P} - \mathbf{C} - R(\mathbf{P} - \mathbf{C} - \langle \mathbf{P} - \mathbf{C}, \mathbf{a} \rangle \mathbf{a}) / \|\mathbf{P} - \mathbf{C} - \langle \mathbf{P} - \mathbf{C}, \mathbf{a} \rangle \mathbf{a}\| \right\| - r \right|.
\end{aligned}$$

## B Derivation of curvature formulae

In Chapter 3 two formulae for the estimation of curvature have been introduced, namely the DN (difference of normals) curvature formula in Equation (3.1) and the interior point formula in Equation (3.6). At this place both formulae are derived from their corresponding geometric configurations involving two points and two surface normals.

Remarkably none of the curvature formulae requires an explicit computation of the centre of a circle or a sphere that approximates a section through a part of a triangulated surface in a certain direction.

### B.1 The DN curvature formula



**Figure B.1:** *The ideal case for the DN curvature formula for two points  $P_1$  and  $P_2$  with normals  $n_1$  and  $n_2$  on a circle*

In the ideal case, it is provided that there are two points  $P_1$  and  $P_2$  on a circle with centre  $M$ , each having a normal  $n_i$  that is perpendicular to the circular curve shown in Figure B.1. Then the curvature  $\kappa$  can be determined simply as the reciprocal value of  $r$  (the radius of the circle) from  $M$  to one of the points as follows. The equation

$$\begin{aligned} \sin \gamma/2 &= \frac{1}{2} \| P_1 - P_2 \| / r \\ &= \frac{1}{2} \| n_1 - n_2 \| \end{aligned}$$

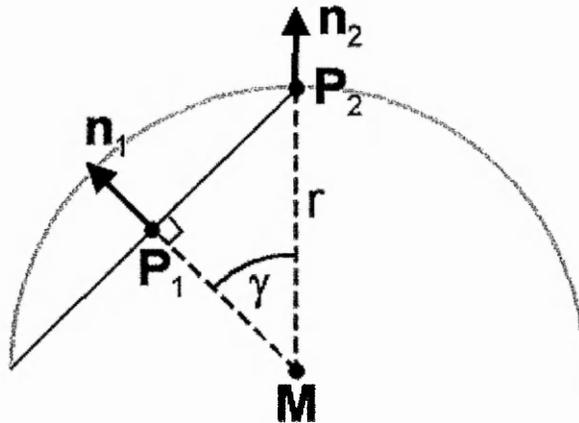
relates the aspect ratios of the triangle  $[M, P_1, \frac{1}{2}(P_1 + P_2)]$  to the triangle  $[0, n_1, \frac{1}{2}(n_1 + n_2)]$  where  $0$  denotes the null vector. This now implies

$$\kappa = 2 \sin(\gamma/2) / \| P_1 - P_2 \|^2$$

$$= \| \mathbf{n}_1 - \mathbf{n}_2 \| / \| \mathbf{P}_1 - \mathbf{P}_2 \|$$

which is the desired result.

## B.2 The interior point formula for a point on a circle and a non-circle point



**Figure B.2:** *The ideal case for the interior point formula for two points  $P_1$  and  $P_2$  with normals  $\mathbf{n}_1$  and  $\mathbf{n}_2$  of which one is on a circle and the other “inside” the circle*

In this ideal case it is supposed that there are two points  $P_1$  and  $P_2$ .  $P_1$  is a point interior to a triangle whereas  $P_2$  is a vertex of a triangle on the circular curve as shown in Figure B.2. Again the normals  $\mathbf{n}_i$  at both points are perpendicular to the circle. In this case the curvature  $\kappa$  can be determined as the inverse of the distance  $r$  from  $M$  to  $P_2$  (similar to the DN curvature formula). By definition of the vector product (e.g. in [Bronstein & Semendjajew 85]) it follows that

$$\begin{aligned} \sin \gamma &= \| \mathbf{P}_1 - \mathbf{P}_2 \| / r \\ &= \| \mathbf{n}_1 \times \mathbf{n}_2 \|. \end{aligned}$$

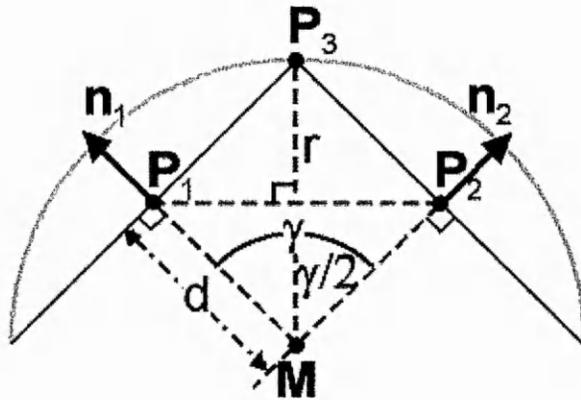
The curvature  $\kappa$  can now be obtained from

$$\kappa = \| \mathbf{n}_1 \times \mathbf{n}_2 \| / \| \mathbf{P}_1 - \mathbf{P}_2 \|.$$

## B.3 The interior point formula for two non-circle points

In the third case ideally there are two points  $P_1$  and  $P_2$  “inside” a circular curve on chords that meet in the point  $P_3$  as shown in Figure B.3. As in the previous case for  $P_1$ , the chords represent the perpendicular cross section of a pair of adjacent

triangles with vertices on the surface. Again the corresponding normals  $\mathbf{n}_i$  at  $\mathbf{P}_1$  and  $\mathbf{P}_2$  are perpendicular to the circle.



**Figure B.3:** *An ideal case for the interior point formula for two points  $\mathbf{P}_1$  and  $\mathbf{P}_2$  with normals  $\mathbf{n}_1$  and  $\mathbf{n}_2$  “inside” a spherical surface*

In this ideal case it is further assumed that the geometry is symmetrical on each side of the line through  $\mathbf{M}$  and  $\mathbf{P}_3$ . It should be noted that  $\mathbf{P}_3$  actually lies a small distance “inside” or “outside” the circular curve. However, the error made by neglecting this distance for the determination of the curvature is in general relatively small compared to the measurement errors in the data points. More precisely, this distance strongly depends on the length of the edge that is shared by the pair of adjacent triangles (which in turn depends on the sampling distance used in order to obtain the scan) and on the radius of the full circle that is tangent to the curve. Moreover, the distance depends on whether the curvature in the perpendicular direction is positive, 0, or negative (e.g. on the surface of a hollow sphere, cylinder, or saddle). However, since a reasonably dense sampling for every part of a surface in 3D space has been presupposed for this project in order to identify surface features, it is justified to neglect this minor error. Furthermore it is a priori unknown whether  $\mathbf{P}_3$  effectively lies “inside” or “outside” the circle since the underlying surface might be concave which would make a compensation of this systematic error unnecessarily complicated and time-consuming.

Now the curvature  $\kappa$  can be estimated as the reciprocal distance  $r$  from  $\mathbf{M}$  to  $\mathbf{P}_3$  as follows:

$$d \sin \gamma/2 = \frac{1}{2} \| \mathbf{P}_1 - \mathbf{P}_2 \|,$$

$$\cos \gamma/2 = d / r,$$

and the equation

$$\sin \gamma = \|\mathbf{n}_1 \times \mathbf{n}_2\|$$

is as in the previous section. This now implies

$$\begin{aligned}\kappa &= 2 \cos(\gamma/2) / d \\ &= \|\mathbf{n}_1 \times \mathbf{n}_2\| / (2 \sin(\gamma/2) d) \\ &= \|\mathbf{n}_1 \times \mathbf{n}_2\| / \|\mathbf{P}_1 - \mathbf{P}_2\|.\end{aligned}$$

## C An alternative method for estimating the characteristic parameters of cylinders, cones, and tori

In Section 4.3 methods are discussed how to obtain estimates for characteristic parameters of geometric primitives from triangulated surface data. Although they have not been used in the present project some alternative estimations for cylinder, cone, and torus are presented in the following.

### C.1 Cylinder

An alternative idea to obtain estimates for the characteristic parameters of a cylinder employs two (not necessarily adjacent) surface triangles with associated compensated centres  $C_1^*$  and normals  $\mathbf{n}_1^*$ . Furthermore, it is assumed to have principal radii  $R_1$  and  $R_2$  for one of the triangles available. To simplify the situation it is additionally assumed that  $R_1$  is the smaller of the two values and that the cylinder is convex (which implies  $R_1 < 0$ ).

Then the axis  $\mathbf{a}$  of the cylinder can be estimated by

$$\mathbf{a} = (\mathbf{n}_1^* \times \mathbf{n}_2^*) / \|\mathbf{n}_1^* \times \mathbf{n}_2^*\|,$$

if  $\|\mathbf{n}_1^* \times \mathbf{n}_2^*\|$  is notably larger than 0. If the length of the vector product is approximately 0, then the compensated centres can be used to estimate the axis unless they are on opposite sides of the cylinder. In this case the axis can be estimated by

$$\mathbf{a} = (C_1^* - C_2^*) / \|C_1^* - C_2^*\|.$$

By shifting one of the compensated centres,  $C_1^*$  say, in the direction of the compensated normal  $\mathbf{n}_1^*$  by the distance of  $R_1$  (in fact,  $R_1 < 0$  implies a shift against this direction) one can obtain a point  $C$  as a point on the axis of the cylinder. More formal,  $C$  is determined by

$$C = C_1^* + R_1 \mathbf{n}_1^*,$$

and  $-R_1$  yields an estimate for the radius  $r$  of the cylinder. If the cylinder is concave, the parameters of the cylinder can be estimated analogously.

Although this technique has not been test in practice, it is expected to give estimates of cylindrical parameters that are as good as the ones presented in Section 4.6.

## C.2 Cone

As for the cylinder, an alternative technique for estimating the characteristic parameters of a cone requires two (not necessarily adjacent) surface triangles with associated compensated centres  $C_1^*$  and normals  $\mathbf{n}_1^*$ . Furthermore, it is assumed to have those principal radii  $R_1$  and  $R_2$  for each of the triangles available that are smaller in magnitude, respectively. Again the situation is simplified by assuming that  $R_1$  is the smaller of the two values and that the cone is convex (such that  $R_1 < R_2 < 0$ ). If this condition cannot be met, it is suggested to repeat the estimation for another pair of triangles.

The idea of the following estimate is to determine (similarly as for the cylinder) for both of the surface triangles a point on the axis of the cone, which can then be connected by a straight line that is in theory parallel to the cone's axis. So the first point  $\mathbf{D}$  on the axis is estimated by

$$\mathbf{D} = C_1^* + R_1 \mathbf{n}_1^*,$$

and the second,  $\mathbf{E}$ , is given by

$$\mathbf{E} = C_2^* + R_2 \mathbf{n}_2^*.$$

Then the axis  $\mathbf{a}$  of the cone can be estimated by

$$\mathbf{a} = (\mathbf{D} - \mathbf{E}) / \|\mathbf{D} - \mathbf{E}\|.$$

The above condition  $R_1 < R_2 < 0$  prevents the points  $\mathbf{D}$  and  $\mathbf{E}$  from coinciding, and thus the denominator of the latter equation is always  $> 0$ . Now the apex  $\mathbf{C}$  of the cone can be estimated similarly to Equation (4.12) by

$$\mathbf{C} = C_1^* + (C_2^* - C_1^*) |R_1| / |R_1 - R_2|.$$

The particular advantage of this alternative method is that it does not need to project any of the compensated centres into a perpendicular intersection plane containing the axis and the surface normal (such as the planes  $Z_1$  and  $Z_2$  in Figures 4.3.1 and 4.3.2). Furthermore, using  $C_2^*$  rather than its projection  $C_2'$  as

in Equation (4.12) is expected to yield more accurate estimates for both apex and opening angle  $2\alpha$ . The latter parameter can be obtained from

$$\alpha = \cos^{-1}(\|C_1^* - C_2^*\| / \|D - E\|).$$

This is because  $C_1^*$  and  $C_2^*$  have approximately the same distance to the surface of the cone, whereas Figure 4.3.3 suggests that  $C_2$  is further away from the surface. However, further investigation is required to confirm the practical use of this method.

### C.3 Torus

As for the cone, a projection plane  $Z$  as in Figure 4.3.5.2 may not be necessary because the point  $D_2$  in Figure 4.3.5.3 can be estimated directly by using the technique of the previous section:

$$D_2 = C_2^* + R_2 n_2^*,$$

where  $R_2$  denotes the principal radius corresponding to  $r_2$  in the latter Figure, and  $n_2^*$  replaces the projected normal  $n_2$ . Again, improved estimates for the characteristic parameters of a torus are expected using this alternative method compared to the method used in Section 4.3.5. However, further research is needed to confirm this.

## **D Publications**

This Chapter includes publications by the Author.

1. Sacchi R, Poliakoff JF, Thomas PD, *Curvature Estimation for Segmentation of Triangulated Surfaces*, Proceedings of the 2<sup>nd</sup> International Conference on 3D Digital Imaging & Modelling, Ottawa, Oct 4-8 1999, pp. 536-543
2. Sacchi R, Poliakoff JF, Thomas PD, Häfele KH, *Improved Extraction of Planar Segments for Scanned Surfaces*, Proceedings of the IEEE Conference on Information Visualization, London, July 19-21 2000, pp. 325-330

# Curvature Estimation for Segmentation of Triangulated Surfaces

R. Sacchi, J.F. Poliakoff, P.D. Thomas

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

K.-H. Häfele

Forschungszentrum Karlsruhe  
Institut für Angewandte Informatik  
Postfach 3640  
76021 Karlsruhe  
Germany

## Abstract

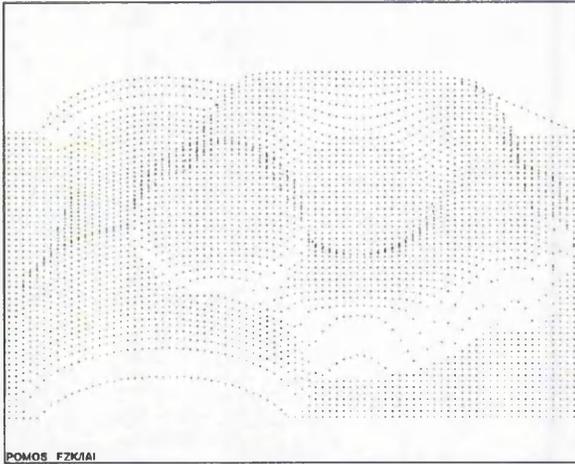
*An important aspect of reverse engineering is the production of digital representations of physical objects for CAD systems. The first stage involves taking 3D coordinate measurements for points on the surface of the object and producing in general an unstructured set of points, called a point cloud. A triangulated surface can be generated from such a point cloud, allowing copies of the original object to be manufactured. However, these triangulated surfaces generally consist of a very large number of triangles with small errors in the positions of their vertices. In many cases the original object is made up of parts of a number of simple geometric objects. Our aim is to segment the triangulated surface into a small number of components, each of which approximates to part of a simple geometric shape. We have developed algorithms for curvature estimation in order to support a 'region growing' method of segmentation.*

## 1 Introduction

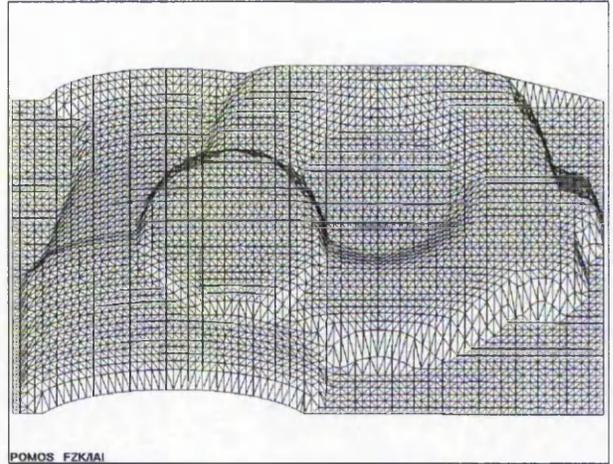
Computer Aided Design (CAD) packages allow engineers to design objects for Computer Aided Manufacturing (CAM) [1]. Subsequent manipulation and modification of the design can be performed relatively easily, because such an object is usually made up of a relatively small number of parts of simple geometric shapes, such as planes, spheres, cylinders, cones and tori. However, very often a physical prototype is produced and then

modified directly or the initial design consists of a physical object to be copied. In such cases a process of reverse engineering is needed, in order to create a digital representation of the object [2,3]. The coordinates of a large number of points on the surface of the object are measured and a point cloud is produced, as shown in Figure 1. At the Research Centre Karlsruhe the POMOS (POint-based MOdelling System) system has been developed to handle large quantities of digitised data, to approximate manually bordered triangulated surfaces with free-form surfaces and to analyse the surfaces generated [4]. Our aim is to extend capabilities of the system to include automated segmentation in order to reduce the manual interaction needed to define surface borders.

Measurement of the coordinates for reverse engineering may involve a tactile method, such as a contact probe, or the process may be non-tactile, for example laser scanning. The resulting data may form a structured point cloud, i.e. it may contain additional information derived from the scanning process. For example, the order of points may be derived from the scanning or triangulation processes can then be simplified. In some cases the points may project onto a grid in, say, the x-y plane, allowing a parametrisation of the points, but in general such a 2½D property cannot be assumed. Previous methods have often relied on the fact that data points were related to such a grid, so-called range data [5].



**Figure 1.** An example of point cloud data with 4018 data points, representing the surface of a technical device.



**Figure 2.** An example of a triangulated surface. The point cloud from Figure 1 has been triangulated, giving 7776 triangles

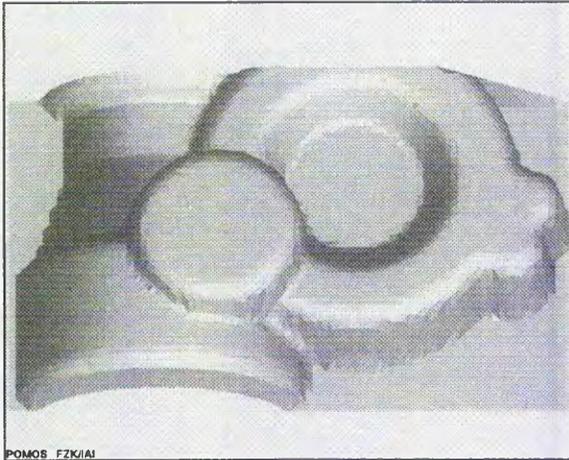
Algorithms have been developed in POMOS for generating a triangulated surface from an unstructured point cloud [6]. Figures 2 and 3 show an example of a triangulated surface obtained from the point cloud shown in Figure 1. Our aim, therefore, is to take such a general triangulated surface and segment it into a small number of simple geometric components. Each of the components must have all its vertices lying, within a given tolerance, on part of a simple geometric shape which can be handled by a CAD system. The geometric shape is allowed to be one of the following: a plane, a sphere, a cylinder, cone or a torus. It is fairly easy for a human operator to identify regions of most of these types for the surface shown in Figures 2 and 3.

In the next sections we give a brief survey of methods of segmentation and curvature estimation, describe our approach to segmentation of a general triangulated surface and explain our method for curvature estimation.

## 2 Segmentation and Curvature Estimation

Previous work in segmentation has been mainly concerned with range data or image data. Previous workers [7,8] have pointed out that the three stages

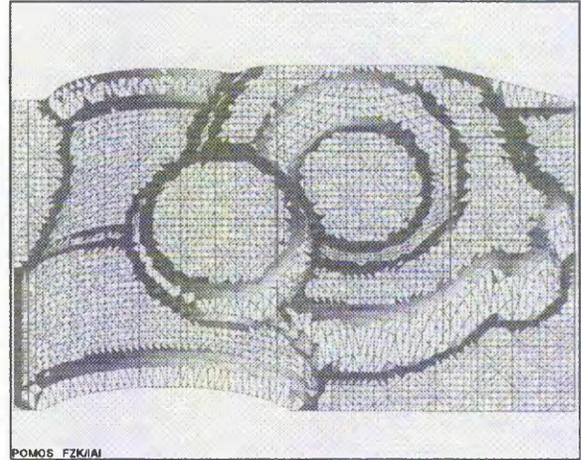
of segmentation, classification and fitting need to be carried out simultaneously rather than sequentially. We refer to the result of the complete process as a segmentation of the surface. Split-and-merge is a top-down method where previous work has exploited for the splitting process the fact that there is a parametrisation available for the data. A bottom-up approach involves starting with a seed point and adding suitable data points until no more can be found [9]. Different regions can be grown in parallel and merged when possible. However, there is the problem of finding suitable seeds initially and the surface shape parameters may need to be adjusted. Curvature has been used to provide preliminary information about surface quality [10]. One approach is to join up points with high curvature in order to attempt to identify ridge lines which can serve as boundaries of surface segments [11,12]. Figure 4 shows regions of high curvature for the surface from Figure 2; some but not all of the joins have been identified. However, smooth joins between, for example, a plane and a cylinder (joined tangentially), cannot be identified in this way, because there is no ridge. Instead there will be a small change in the curvature itself which cannot



**Figure 3.** The triangulated surface from Figure 2 with the triangles shaded and illuminated, showing the shape more clearly.

be detected reliably with noisy data. Various clustering methods have been applied successfully to range data but many rely on the parametrisation available from such data [13]. Other stochastic processes are computationally intensive [14] and so are unsuitable for the large number of data points which usually occur in reverse engineering.

We have found that the region growing approach is the most appropriate for general triangulated surfaces. Suitable 'seed' triangles are chosen and then further triangles are added provided that they approximate to the required geometric shape. Readjustment of shape parameters can be achieved by geometrical or other optimisation methods. In order to identify suitable seed triangles for region growing, we have developed algorithms for estimation of the curvature associated with a given triangle. A surface made up of triangles has zero curvature at all points where the curvature is defined (i.e. in the interior of a triangle). However, we are concerned with the triangles as representing the topology (or neighbourhood information [3]) of the measured surface, i.e. the connections between the vertices, which are the actual measured data. Therefore we associate with each triangle a



**Figure 4.** The shaded region shows triangles of high estimated mean curvature from the surface in Figure 2 (mean curvature values above 0.1). Most occur where we would expect to find boundaries between segments but not all boundaries appear.

curvature estimate intended to approximate to the curvature of the measured surface, assuming that it is smooth. If the triangle corresponds to a smooth part of the surface, then the curvature estimate is likely to approximate to the curvature of that surface. If the triangle corresponds to a part of the surface which is not smooth, such as an edge, then the curvature estimate will tend to be much larger in magnitude but may not be close to that of the surface. This will not cause a problem, since we use triangles with low or medium estimated curvature to seed the region growing process.

The curvature of a surface can be found using analytic methods using derivatives but these cannot be applied to digitised data directly and require the fitting of a smooth surface to some of the data points. Flynn et al. [15] proposed an algorithm for estimating the curvature between two points on a surface which uses the surface normal change between the points. Krsek et al. [16] discuss the angular deficit at a vertex as a measure of the curvature but this may have problems with noisy data. Another method is to use an estimate of the

angular variation of the normal close to a particular vertex [2]. However the last two methods, although they are related to the curvature, do not measure it directly. In this paper we propose a method related to that of Flynn et al. but using normals which have been compensated to minimise the effect of small errors in the data.

### 3 Our Approach to Segmentation

We assume that the triangulated surface consists of a set of triangles with the following properties:

- (i) Each triangle has non-collinear vertices;
- (ii) Any edge belongs to at most two triangles;
- (iii) No two triangles intersect except in one edge or one vertex;
- (iv) Any pair of triangles can be connected by an edge-connected sequence, i.e. a sequence of triangles containing both triangles for which each triangle in the sequence shares an edge with the next triangle. (Thus, for example, two disconnected components or two components joined by only a single vertex are not allowed.)

Any subset of a triangulated surface will itself be a triangulated surface provided that it satisfies (iv). We call a surface closed if each edge belongs to exactly two triangles. Otherwise it is open and we call triangles internal to the surface, if they have three adjacent triangles.

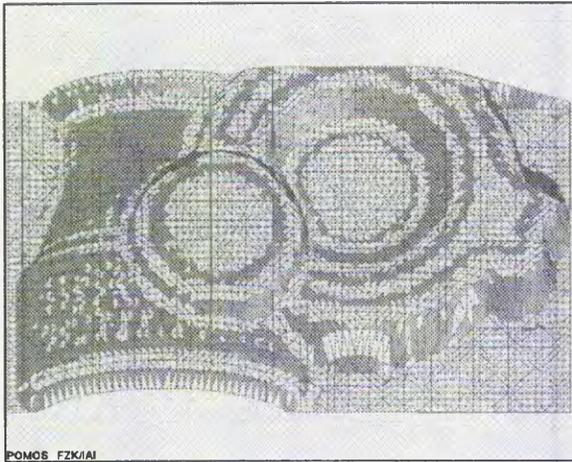
For a segmentation of the triangulated surface we require the decomposition of the set of triangles into a number of disjoint components, each of which satisfies (iv) above, such that the following are satisfied:

- A. Each component approximates, where possible, to part of a particular geometric primitive, i.e. all of its vertices lie within a given tolerance of the geometric primitive;

- B. No two components associated with the *same* geometric primitive with the *same* parameters can share a whole edge. (Otherwise they could be combined into one component.)
- C. The number of components is as small as possible for the given tolerance.
- D. The geometric primitives allowed are: planes, spheres, cylinders, cones and tori.

Obviously, without condition C there will always be a segmentation into planes, one for each triangle of the surface, since each triangle belongs to the plane defined by its three vertices. This can be done even if the tolerance is set to zero. However, we are seeking a segmentation into a small number of components, grouping many triangles together. Thus we include condition C. We could require components to be as large as possible but in some cases boundaries may need to be adjusted where there is a conflict when triangles belong to more than one maximal component. Thus, it can also be seen that, in general, conditions A - C do not determine a unique segmentation.

We have implemented planar segment extraction based on the curvature estimation method described in the next section. Region growing starts with a triangle of low estimated mean curvature, which forms the initial region. Repeatedly an attempt is made to add to the region a new triangle which shares an edge with a triangle already in the region, thus adding a single new vertex. This process continues until no more such triangles can be added. A triangle is allowed to be added, if its vertices lie within the given tolerance of the plane associated with the region. If that is not possible, then it is added temporarily to the region and an attempt is made to adjust the parameters of the plane so that *all* the triangles of the new region lie within the required tolerance of the new plane. When this can be done the triangle is added and the parameters are adjusted to those of the new plane.

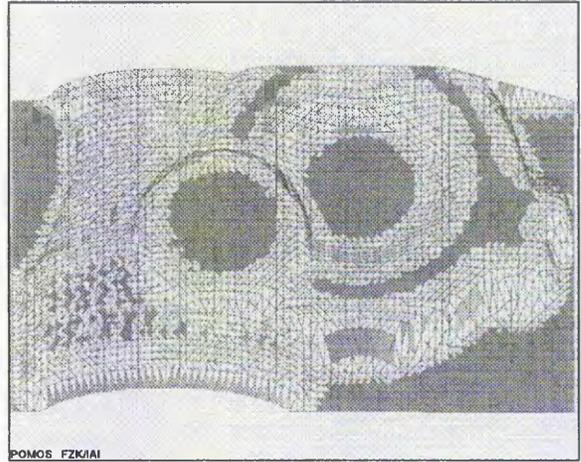


**Figure 5.** The shaded region shows triangles of medium estimated mean curvature from the surface in Figure 2 (mean curvature values between 0.03 and 0.1).

#### 4 Curvature Estimation for General Triangulated Surfaces

We assume that we have a triangulated surface, as above. In order to give a sign to a curvature value, we need to have a normal direction defined for each triangle. Effectively we need to know which side of the triangle faces towards the outside of the object, because it is not difficult to find a line perpendicular to a plane defined by three points. It is essential that the assignment of normals is consistent over the whole surface, i.e. any two triangles sharing an edge have consistent normal directions. Previous work in POMOS [6] also produced a method for providing a consistent assignment of the directions for the triangle normals. Therefore we consider a triangulated surface satisfying (i) to (iv) above which has a consistent set of normals and estimate the curvature for every triangle. In order to simplify the description, we assume that the triangle is internal to the surface but the method can easily be extended to all triangles.

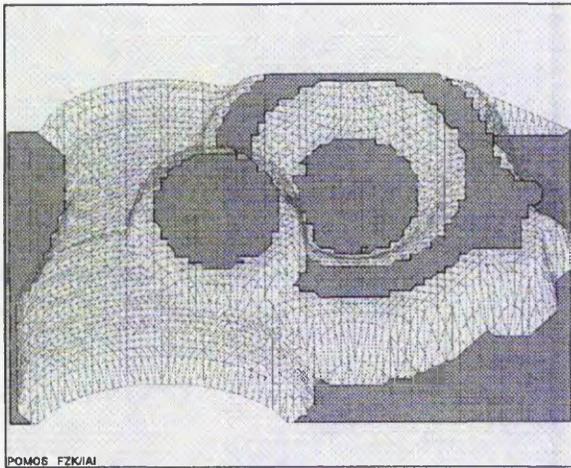
For any pair of triangles which share an edge we can find the curvature of the sphere passing through the four vertices involved, unless they are coplanar



**Figure 6.** The shaded region shows triangles of low estimated mean curvature from the surface in Figure 2 (mean curvature values below 0.03). Most of them occur where we would expect planar segments.

in which case the curvature is zero. The sign of the curvature is taken as positive if the centre of the sphere is on the same side of the surface as the two normals, and negative otherwise. An estimated curvature value for a given triangle can then be defined as the average of the curvatures obtained when it is paired with each of the three adjacent triangles in turn. However, small errors in the data can affect the results obtained. We describe below an improved method for finding an estimate of the curvature for a pair of triangles, which we use to derive a method for estimating the mean curvature associated with a triangle.

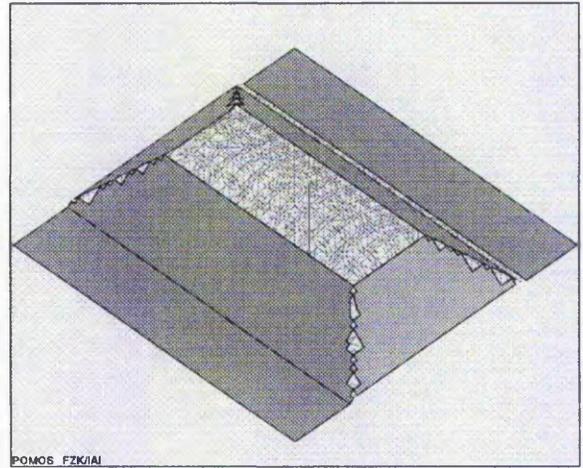
In order to compensate for the effect of errors in the positions of the triangle vertices, we replace the normal for each triangle by a 'compensated normal', as follows. Firstly, we estimate a normal for each vertex, which we call an 'interpolated normal', equal to the weighted average of the normals for all triangles meeting at that vertex. The weighting used for each normal is the area of the triangle. We now take as the compensated normal for a triangle the weighted average of the three interpolated normals at the vertices of the triangle,



**Figure 7.** The six planar segments identified on the surface from Figure 2, using region growing with seed triangles of low curvature and tolerance 0.009.

using as weighting factor, for each vertex, the sum of the areas of the triangles meeting at that vertex. In a similar way, we define the 'compensated centre' of each triangle as the weighted average of the vertices using the same weighting factors. Then, for a pair of triangles with compensated centres  $\mathbf{c}_1$  and  $\mathbf{c}_2$  and compensated normals  $\mathbf{n}_1$  and  $\mathbf{n}_2$ , respectively, we estimate the curvature as  $\|\mathbf{n}_1 \times \mathbf{n}_2\| \square \|\mathbf{c}_1 - \mathbf{c}_2\|$ . This differs from the formula  $\|\mathbf{n}_1 - \mathbf{n}_2\| \square \|\mathbf{c}_1 - \mathbf{c}_2\|$  corresponding to that given by Flynn et al. [15], because we allow for the fact that, in our case,  $\mathbf{c}_1$  and  $\mathbf{c}_2$  are not on the surface of the supposed sphere but inside it. For a given triangle we now have three curvature values. In a similar way, we estimate another three values by pairing the compensated normal with the interpolated normal at each of the three vertices in turn. Finally, for the estimated mean curvature we take the average of the maximum and minimum of the six curvature estimates obtained for that triangle.

We have found that the method proposed by Flynn et al. [15] to find the sign of the curvature needs to be modified. In our terminology, they assign the curvature to be positive, for cases where  $\|\mathbf{c}_1 - \mathbf{c}_2\| \square \|\mathbf{c}_1 + \mathbf{n}_1 - (\mathbf{c}_2 + \mathbf{n}_2)\|$ , and to be negative



**Figure 8.** Another example where six planar regions have been identified for synthetic data with planar and cylindrical regions.

otherwise. However, this method can give the wrong result when  $\|\mathbf{c}_1 - \mathbf{c}_2\|$  is much smaller than 1, resulting in a false positive for a region of negative curvature. This happens in such cases, because the normals are much larger in magnitude than  $\|\mathbf{c}_1 - \mathbf{c}_2\|$ . We avoid this problem by scaling the normals, i.e. by replacing each  $\mathbf{n}_i$  by  $(\frac{1}{2}\|\mathbf{c}_1 - \mathbf{c}_2\|)\mathbf{n}_i$ , before applying the criterion. It should also be noted that our method for estimating the normal at a vertex is similar to that of Hoschek et al. [17]. By contrast, we use the area of the triangle rather than its inverse as weighting; we have found that, in general, the smaller the triangle area the greater the effect on the normal of errors in vertex positions.

## 5 Results

Curvature estimation and planar segment extraction have been implemented within the POMOS system. Figures 4, 5 and 6 show regions of high, medium and low mean curvature, respectively, for the surface shown in Figure 2. It can be seen that some triangles in parts that we expect to be planar have medium or even high curvature because of noise in the data. However triangles which are not close to the boundaries of a

'planar' region normally have low curvature. On the other hand, noisy data causes some low curvature triangles to be found in parts where we do not expect to find planes but these do not result in planar segments. Figure 7 shows the results for planar segment extraction using as seeds triangles of low curvature, as shown in Figure 6. It can be seen that the planar regions found correspond well to what we expected. However these regions include some triangles of medium or even high curvature, especially near their boundaries. Figure 8 shows the results of planar extraction using synthetic data with planar and cylindrical regions.

## 6 Conclusions

Preliminary results show that the curvature estimation method successfully allows suitable triangles to be identified in order to seed the region growing process for planar segment extraction. Although some of the estimates are inappropriate because of noise in the data, the region growing process can identify underlying planar regions where they exist. Work is in progress on the extraction of the other geometric primitives.

## Acknowledgements

We thank the Nottingham Trent University and the Research Centre Karlsruhe for supporting this work and for providing facilities for the research.

## References

- [1] Hoschek J, Lasser D, *Fundamentals of Computer Aided Geometric Design*, AK Peters, Wellesley, Massachusetts, 1993.
- [2] Hoschek J, Dietz U, Wilke W, "A Geometric Concept of Reverse Engineering of Shape: Approximation and Feature Lines", in: Daehlen M, Lyche T, Schumaker LL (Eds.), *Mathematical Methods for Curves and Surfaces II*, pp. 253-262, 1998.
- [3] Varady T, Martin RR, Cox J, "Reverse Engineering of Geometric Models – An Introduction", *Computer-Aided Design*, Vol. 29 No. 4, pp. 255-268, 1997.
- [4] Häfele KH, "POMOS – POInt Based MOdelling System", in: Hoschek, J, Dankwort, W (Eds.), *Reverse Engineering*, Verlag B.G. Teubner, Stuttgart/Germany, 1996.
- [5] Jiang XY, Bunke H, "Fast Segmentation of Range Images into Planar Regions by Scan Line Grouping", *Machine Vision & Applications*, Vol. 7, No. 2, pp 115-122, 1994.
- [6] Häfele KH, Hellmann M, "3D-Meßdatenaufbereitung und Weiterverarbeitung mit POMOS", 2. Workshop Optische 3D-Formerfassung Großer Objekte, Esslingen, Germany, 1996.
- [7] Besl PJ, Jain RC, "Segmentation Through Variable-Order Surface Fitting", *IEEE Transactions on Pattern Analysis & Machine Intelligence*, V. 10, No. 2, pp. 167-192, 1988.
- [8] Taubin G, "Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation", *Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 11, pp. 1115-1138, 1991.
- [9] Maître G, Hügli H, Tièche F, Amann JP, "Range Image Segmentation Based on Function Approximation", in: A Gruen & E Baltsavias (Eds.), *Close-Range Photogrammetry Meets Machine Vision*, SPIE Vol. 1395, pp. 275-282, 1990.
- [10] Trucco E, Fisher RB, "Experiments in Curvature-Based Segmentation of Range Data", *IEEE Transactions on Pattern Analysis & Machine Intelligence*, Vol. 17, No. 2, pp. 177-182, 1995.
- [11] Chen YH, Liu CY, "Robust Segmentation of CMM Data Based on NURBS", *International Journal of Advanced Manufacturing Technology*, Vol. 13, pp 530-534, 1997.

- [12] Lukács G, Andor L, "Computing Natural Division Lines on Free-Form Surfaces Based on Measured Data", in: Daehlen M, Lyche T, Schumaker LL (Eds.), *Mathematical Methods for Curves and Surfaces II*, pp. 319-326, 1998.
- [13] Hebert M, Ponce J, "A New Method for Segmenting 3-D Scenes into Primitives", *Proc. International Conference on Pattern Recognition*, Munich, pp. 836-838, 1982.
- [14] Roth G, Levine MD, "Extracting Geometric Primitives", *Computer Vision, Graphics & Image Processing - Image Understanding*, Vol. 58, No. 1, pp. 1-22, 1993.
- [15] Flynn PJ, Jain AK, "On Reliable Curvature Estimation", *IEEE Conference on Pattern Recognition, San Diego*, pp. 110-116, 1989.
- [16] Krsek P, Lukács G and Martin RR, "Algorithms for Computing Curvatures from Range Data", in: R. Cripps (Ed.), *The Mathematics of Surfaces VIII*, pp.1-16, 1998.
- [17] Hoschek H, Dietz U, Wilke W, "A Geometric Concept of Reverse Engineering of Shape: Approximation and Feature Lines", in: Daehlen M, Lyche T, Schumaker LL (Eds.), *Mathematical Methods for Curves and Surfaces II*, pp. 253-262, 1998.

# Improved Extraction of Planar Segments for Scanned Surfaces

R. Sacchi, J.F. Poliakoff, P.D. Thomas  
*Department of Computing*  
*The Nottingham Trent University*  
*Burton Street, Nottingham*  
*NG1 4BU*  
*England*  
*jfp@doc.ntu.ac.uk, pdt@doc.ntu.ac.uk*

K.-H. Häfele  
*Forschungszentrum Karlsruhe*  
*Institut für Angewandte Informatik*  
*Postfach 3640*  
*D-76021 Karlsruhe*  
*Germany*  
*haefe@iai.fzk.de*

## Abstract

*The reverse engineering of a physical object often requires the production of a digital representation of the object. The object surface is scanned and a large number of points are obtained. These points are often organised in some way which provides adjacency information. However, in other cases, the result is an unstructured set of points, or point cloud. From such a point cloud a triangulated surface can be generated, so in all cases adjacency information can be obtained. Copies of the original object can be manufactured using the triangulated surface to define the shape. Often the original object is made up of parts of a number of simple geometric primitives and could be represented much more simply. Our aim is to segment a discretely represented surface into a small number of such simple geometric components using a 'region growing' approach. This paper describes an improved algorithm (with 'super triangles') for planar extraction, which requires only the data points and some form of adjacency information.*

## 1. Introduction

Engineers use Computer Aided Design (CAD) packages to design objects to be produced by Computer Aided Manufacturing (CAM) systems [1,2]. Such objects are usually made up of fairly small numbers of parts of simple geometric shapes, so they can be relatively easily manipulated, modified and manufactured. However, in some cases a physical prototype is produced and is then modified directly; in others, the initial design consists of a physical object to be copied. In many cases, a process of reverse engineering is needed in order to create a digital representation of an object [3]. The positions of a large number of points on the surface of the object are

measured. We aim to develop automated segmentation of such surfaces into simple geometric parts.

The data points for reverse engineering may be obtained by a tactile method, such as a contact probe, or the process may be non-tactile, for example laser scanning. The resulting data may form a structured point cloud, i.e. it may contain adjacency information derived from the scanning process, such as the order of the points. In some cases, the points may project onto a grid in, say, the  $x$ - $y$  plane, which then allows the points to be parametrised. In general, however, such a 2½D property cannot be assumed. Previous methods for segmentation have often relied on the fact that data points were related to such a grid, so-called range data [5] or image data based on pixels.

When the data points form an unstructured point cloud, it is nevertheless possible to generate a triangulated surface from it [6,7,8]. Our aim is to take such a general surface which has some form of adjacency information and to segment it into a small number of simple geometric components. Each of the components must have all its data points lying, within a given tolerance, on part of a simple geometric shape which can be handled easily by a CAD system. The geometric shape is assumed to be one of the following: a plane, a sphere, a cylinder, cone or a torus. It is relatively easy for a human operator to classify regions into one of these types for the surfaces shown in Figures 1, 3, 4 and 5. In the next sections we give a brief survey of methods of segmentation and explain our algorithm for fast planar extraction using data points and adjacency information alone.

At the Research Centre Karlsruhe, the POMOS (POint-based MOdelling System) system has been developed to handle large sets of digitised data. It is also able to triangulate unstructured 2½D data, to approximate manually segmented triangulated surfaces with free-form surfaces and to analyse the surfaces generated [4]. We

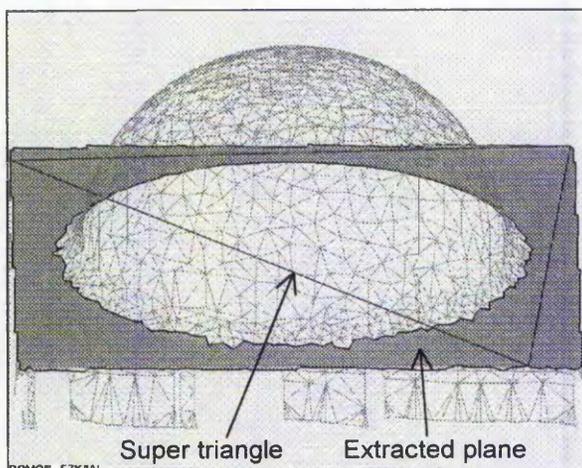


Figure 1. An example of a triangulated surface where, as expected, a single planar segment (shaded) has been found in 2 sec. The final super triangle has been superimposed. (3,473 points, tolerance 0.0005, adjustment tolerance 30%, minimum triangles 150.)

have used this system as a platform to develop and test our algorithms.

## 2. Segmentation

Previous work in segmentation has been mainly concerned with range data or image data [9,10]. Split-and-merge is a top-down method for which the splitting process has most often relied on the parametrisation of the data. A bottom-up approach involves starting with a seed point and adding suitable data points until no more can be found [11]. Different regions can be grown and then merged when possible. During the growing process, surface shape parameters may need to be adjusted. Curvature has been used to provide preliminary information about surface quality [5]. One approach to segmentation is to attempt to join up points where curvature is high in order to identify ridge lines which form the boundaries of surface segments [12]. Some boundaries can be found in this way. However, a smooth join between two segments, such as a plane and a cylinder joined tangentially, cannot be identified in this way, because there is no ridge. There will often be a small change in the curvature itself but this cannot be detected reliably with noisy data. Clustering methods, such as the Hough Transform, have been applied successfully to range data, but many rely on the parametrisation of the data [13]. Genetic algorithms for primitive extraction are computationally intensive [14], so they are unsuitable for the complex objects which usually occur in reverse engineering.

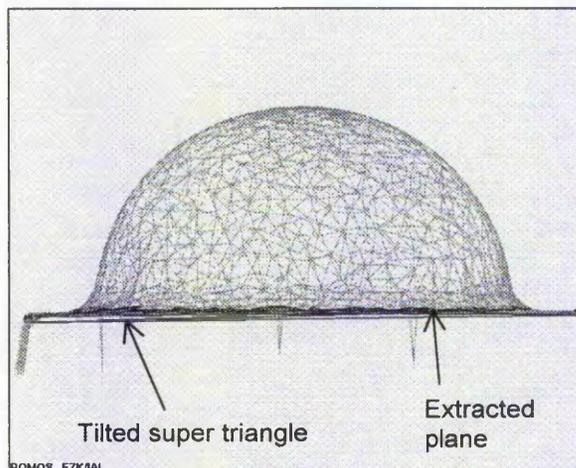


Figure 2. The data from Figure 1 has been processed again but with the adjustment tolerance equal to the given tolerance. In the side view it can be seen that the final super triangle is now tilted. Three planar segments are found instead of one (Time taken 13 sec.)

We have found that the region growing approach is the most appropriate for general triangulated surfaces [1]. Suitable 'seed' points are chosen and then further adjacent points continue to be added while they approximate to the required geometric shape. Readjustment of shape parameters is needed during the growing process. We have developed a fast method for planar extraction which uses region growing. This method can also be applied to general surface data, provided that it contains some sort of adjacency information.

Ashbrook et al. [15] have explained that planar patches are very important, because many mechanical objects are made up of planes. In 1982 Hebert et al. [13] described a method for extracting primitives using the Hough transform but it is time-consuming and memory-intensive. Since then other methods have been proposed for extraction of planar segments range images but they rely on the parametrisation [16,17,18].

We have previously described in detail our approach to segmentation of triangulated surfaces using region growing [1]. The method is based on the curvature estimation algorithm which we had developed. Region growing starts with a triangle of low estimated mean curvature, which forms the initial region. However, we have found that for planar segments the time taken for curvature estimation is longer than the time saved in avoiding false starts. Without curvature estimation, the method can be adapted immediately to the more general case. Our new algorithm, described in Section 3.2, was developed using an idea proposed by Roth et al. [14] for representing a geometric primitive by an appropriate minimal set of points.

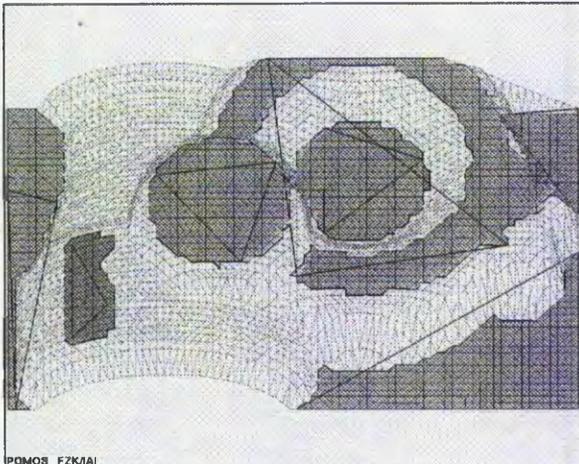


Figure 3. Seven planar segments, six of which resemble planes, have been found in 25 sec. One segment appears to be part of a cylinder of large radius. (4.018 points, tolerance 0.279, adjustment tolerance 50%, minimum triangles 100.)

### 3. Fast extraction of planar segments

We first describe a simple region growing algorithm for planes and then explain how our algorithm improves on such methods.

#### 3.1. Simple region growing

The simple region growing process starts with a 'seed' region of three non-collinear adjacent points which therefore define an initial plane. Repeatedly an attempt is made to add a new point to the region. The candidate point must be adjacent to a point already in the region. The process continues until no more such points can be added. When a point is considered for addition, one of two cases occurs:

- (i) it lies within the given tolerance of the plane associated with the current region;
- (ii) it is not within the tolerance of the plane.

In case (i) the point is added immediately to the region. In case (ii) it is added temporarily to the region and an attempt is made to adjust the plane so that *all* the points of the new region lie within the required tolerance of the new plane. When this can be done the candidate point is added to the region and the plane is updated.

In order to prevent very small planar regions from being extracted, an additional parameter is used, the minimum number of points for a region. (We have used a minimum number of triangles in our implementation.)

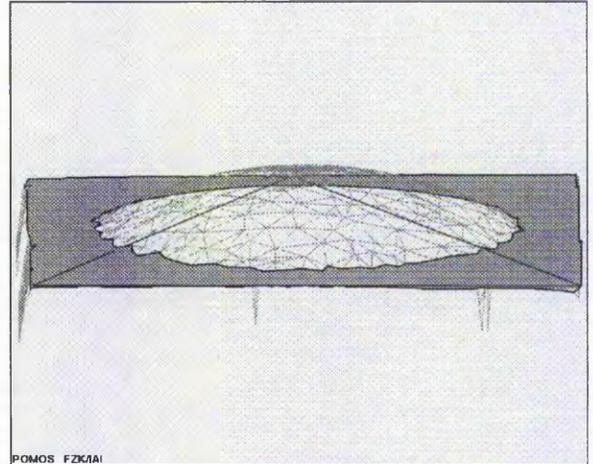


Figure 4. An example where, as expected, a single planar segment has been found in 10 sec. (2,750 points, tolerance 0.00068, adjustment tolerance 50%, minimum triangles 150.)

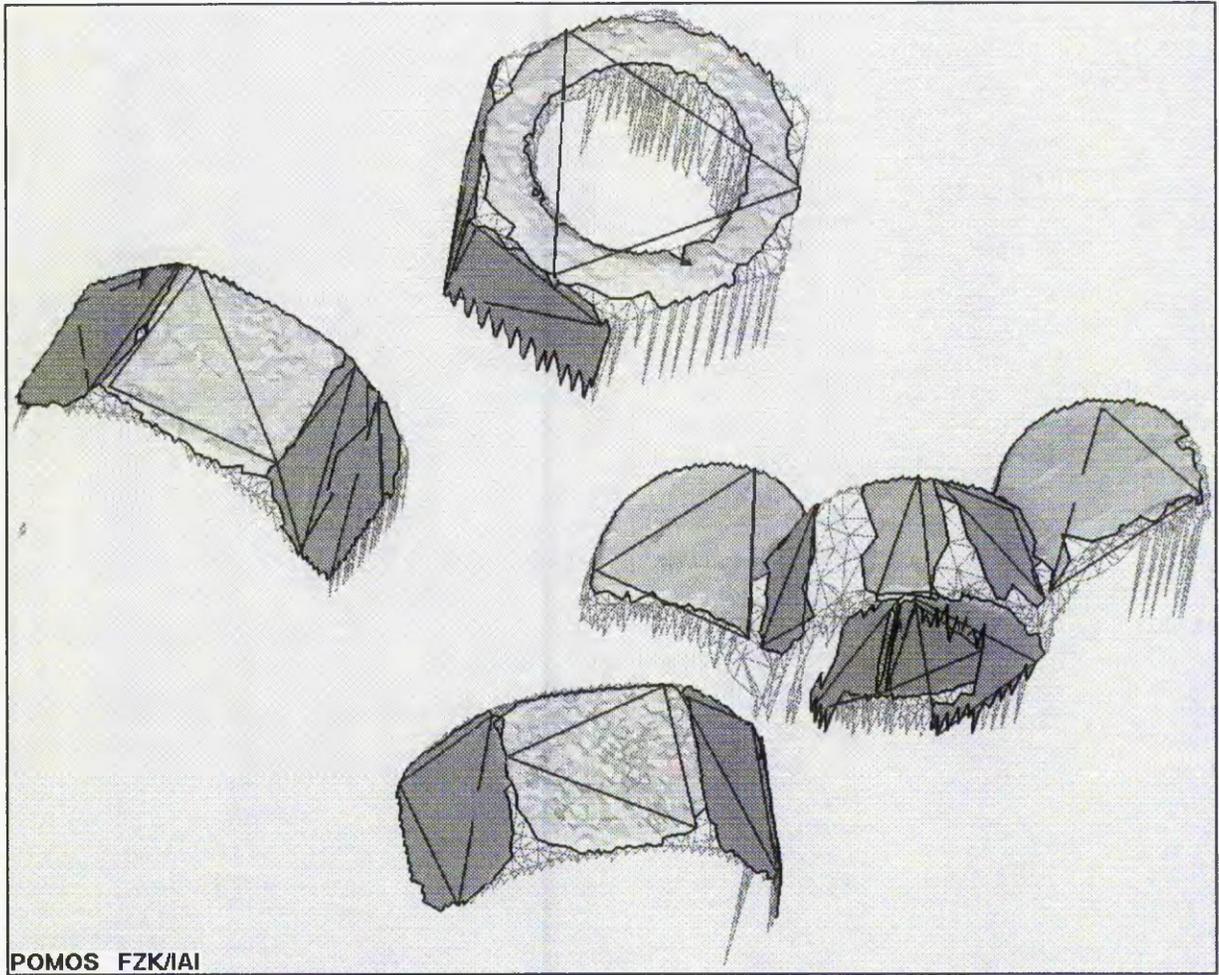
It would be possible to adjust the plane every time a point is added in order to achieve a new plane which best fits the data points. However, the region growing process will then take much longer. Our new algorithm speeds up the process even more.

#### 3.2. Region growing with super triangles

The simple algorithm from section 3.1 can be modified as described below.

We have found that the region growing can be made even faster by restricting the possible planes to those defined by any triangle formed from any set of three points within the current region. Such a triangle we call a 'representative triangle' for the plane. We attempt to grow this representative triangle by making its area as large as possible. The idea is that for reasonably large regions there will only be a small 'tilt' to the plane caused by adjusting the plane when another point is added. This restriction may cause failure of some seeds to grow successfully. However, for any region which is approximately planar there will be many possible seed regions available.

At each stage of the region growing process therefore, the current region is associated with a current plane defined by a representative triangle (with vertices among the points in the current region). There is also associated a 'super triangle' of area at least as large as that of the representative triangle. Initially the two triangles are both defined by the three non-collinear points of the seed region.



POMOS FZK/IAI

Figure 5. Twenty planar segments have been found in 23 sec. for this example of a scan of various nuts in different positions. (3,666 points, tolerance 0.0005, adjustment tolerance 50%, minimum triangles 20.)

Now, if case (i) occurs, the new point is added to the region and the super triangle is modified using the new point, as described below. However, the representative triangle (and therefore the plane) is left unchanged.

In case (ii), we consider the plane defined by the super triangle. If *all* the points (including the candidate point) lie within an 'adjustment tolerance' of this plane, it becomes the new plane and the candidate point is added to the region. The super triangle thus becomes the new representative triangle and the super triangle is modified (see below). Therefore the area of the representative triangle will never decrease and will tend to increase, because of the way the super triangle is found. Figure 1 shows an example where a single planar segment has been extracted together with the final super triangle.

We now describe how a new super triangle is found (in such a way that the area will tend to increase). The new triangle is generated from the old one using both the

new point and the previous point added. An attempt is made to replace the old triangle by one of larger area by checking all possible triangles with vertices chosen from the above two points together with the old triangle's three vertices. If an increase in area is possible, the replacement is chosen as such a triangle with maximum area. Otherwise, the old triangle is retained.

We have found that, if the adjustment tolerance has the *same* value as the given tolerance, then case (ii) can lead to a considerable tilting of the super triangle, as shown in Figure 2. This can prevent region growth and result in the splitting of one approximately planar part into several planar regions. Therefore we have used a smaller value than the given tolerance, which reduces the tilt without having much effect on the size of region grown. The case shown in Figure 2 produces three planar segments, whereas in Figure 1 with adjustment tolerance of 30% gives a single planar segment for the same data and takes

a shorter time. For the examples in Figures 3, 4 and 5 the adjustment tolerance was set to 50% of the given tolerance.

We have also found that many data sets are derived from points scanned in a systematic way. For such data sets the results are improved, if an attempt is made to randomise the order in which the points are chosen as seed points. Without this 'pseudo-randomisation' the seed point tends to be one near the boundary of the region and the initial plane is sometimes a bad approximation to the plane. This can cause smaller regions to grow and again result in splitting of regions.

## 4. Results

Fast planar segment extraction has been implemented within the POMOS system on a Silicon Graphics O2 workstation. Figures 1, 3, 4 and 5 show the results of planar segment extraction for various triangulated surfaces using adjacency information alone. It can be seen that the planar regions found generally correspond well to what we expected. However, in Figure 3 an additional region has grown on a part resembling a cylinder of large radius. This can be avoided by increasing the value of the minimum triangles parameter but then small, genuinely planar, segments may be missed. In Figure 4 the process has successfully distinguished between the planar part and slightly raised dome.

## 5. Conclusions

Preliminary results show that the new algorithm is successful in achieving planar segment extraction when a suitable value of tolerance is provided. Work is in progress to develop a method for estimating a suitable tolerance value automatically. The current algorithmic performance is by no means optimal, because of the data structure used in POMOS. On a dedicated system we expect that the computational time would be noticeably lower.

## 6. Acknowledgements

We thank the Nottingham Trent University and the Research Centre Karlsruhe for supporting this work and for providing facilities for the research. We are also grateful to the National Research Council of Canada and the Technische Hochschule Darmstadt (Germany) for making available suitable test data.

## 7. References

- [1] Sacchi R, Poliakoff JF, Thomas PD, Haeefele K-H, "Curvature Estimation for Segmentation of Triangulated Surfaces", *Proc. 2<sup>nd</sup> Int. Conference on 3-D Digital Imaging and Modeling*, Ottawa, Canada, pp. 536-544, 1999.
- [2] Hoschek J, Lasser D, *Fundamentals of Computer Aided Geometric Design*, AK Peters, Wellesley, Massachusetts, 1993.
- [3] Várady T, Martin RR, Cox J, "Reverse Engineering of Geometric Models - An Introduction", *Computer-Aided Design*, Vol. 29 No. 4, pp. 255-268, 1997.
- [4] Häfele KH, "POMOS - POINT Based MOdelling System", in: Hoschek, J, Dankwort, W (Eds.), *Reverse Engineering*, Verlag B.G. Teubner, Stuttgart/Germany, 1996.
- [5] Besl PJ, Jain RC, "Segmentation Through Variable-Order Surface Fitting", *IEEE Transactions on Pattern Analysis & Machine Intelligence*, V. 10, No. 2, pp. 167-192, 1988.
- [6] Häfele KH, Hellmann M, "3D-Meßdatenaufbereitung und Weiterverarbeitung mit POMOS", 2. Workshop Optische 3D-Formerfassung Großer Objekte, Esslingen, Germany, 1996.
- [7] Hoppe H, *Surface Reconstruction from Unorganised Points*, PhD Thesis, University of Washington, 1994.
- [8] Yemez Y, Schmitt F, "Progressive Multilevel Meshes from Octree Particles", *Proc. 2nd Int. Conference on 3D Digital Imaging and Modeling*, Ottawa, pp. 290-299, 1999.
- [9] Biswas PK, Biswas SS, Chatterji BN, "An SIMD Algorithm for Range Image Segmentation", *Pattern Recognition*, Vol. 28, No.2, pp. 255-267, 1995.
- [10] Leonardis a, Gupta A, Bajcsy R, "Segmentation of Range Images as the Search for Geometric Parametric Models", *Int. Journal of Computer Vision*, Vol. 14, No. 3, pp. 253-277, 1995.
- [11] Maitre G, Hügli H, Tièche F, Amann JP, "Range Image Segmentation Based on Function Approximation", in: A Gruen & E Baltsavias (Eds.), *Close-Range Photogrammetry Meets Machine Vision*, SPIE Vol. 1395, pp. 275-282, 1990.
- [12] Lukács G, Andor L, "Computing Natural Division Lines on Free-Form Surfaces Based on Measured Data", in: Daehlen M, Lyche T, Schumaker LL (Eds.), *Mathematical Methods for Curves and Surfaces II*, pp. 319-326, 1998.
- [13] Hebert M, Ponce J, "A New Method for Segmenting 3-D Scenes into Primitives", *Proc. International Conference on Pattern Recognition*, Munich, pp. 836-838, 1982.
- [14] Roth G, Levine MD, "Geometric Primitive Extraction using a Genetic Algorithm", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 9, pp. 901-905, 1994.
- [15] Ashbrook AP, Fisher RB, Robertson C, Werghi N, "Segmentation of Range Data into Rigid Subsets using Planar Surface Patches", *Proc. British Machine Vision Conference*, Essex pp. 530-539, 1997.
- [16] Jiang XY, Bunke H, "Fast Segmentation of Range Images into Planar Regions by Scan Line Grouping", *Machine Vision & Applications*, Vol. 7, No. 2, pp 115-122, 1994.

[17] Faugeras OD, Hebert M, Pauchon E, "Segmentation of Range Data into Planar and Quadric Patches", *Proc. 3<sup>rd</sup> Conference on Computer Vision and Pattern Recognition*, Arlington, VA, pp. 8-13, 1983.

[18] Taylor RW, Savini M, Reeves AP, "Fast Segmentation of Range Imagery into Planar Regions", *J. Computer Vision, Graphics and Image Processing*, Vol. 45, No. 1, pp. 42-60, 1989.

## **E Hardware used for this project**

The figures in Chapter 3 and Chapter 4 which have been produced using the POMOS software tool involve the following hardware configuration:

SGI Indigo 2 Extreme Workstation  
IRIX 6.2 Operating System  
R4400 Processor with 192 MB Ram

The figures in Chapter 1 and Chapter 5 which have been produced using the POMOS software tool involve the following hardware configuration:

SGI O2 Workstation  
IRIX 6.5 Operating System  
R5000 Processor with 320 MB Ram