VISUALIZATION AND MODELING CONTOURS OF TRIVARIATE FUNCTIONS

by

Bernd Hamann

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

ARIZONA STATE UNIVERSITY

August 1991

VISUALIZATION AND MODELING CONTOURS

OF TRIVARIATE FUNCTIONS

by

Bernd Hamann

has been approved

May 1991

APPROVED:

Gregory M. Nielson , Chairperson

Robert E. Barnhill

Gerald E. Farin

Ben M. Huey

Thomas A. Foley

Hans Hagen

Supervisory Committee

ACCEPTED:

Robert E. Barnhill
Department Chairperson

Neil Hadley
Dean, Graduate College

Abstract

In many real world applications one is concerned with the problem of visualizing and approximating three-dimensional data, commonly referred to as scalar fields (points in three-dimensional space with associated function values). The data themselves can either be physical measurements or be obtained from a mathematical simulation. Typical applications are found in medicine (computerized axial tomography, magnetic resonance imaging), in geology and meteorology (temperature, pressure, radiation), and in the CAD/CAM industry (car body, ship hull, airplane design).

Based on existing methods for visualizing bivariate functions new techniques are presented for rendering three-dimensional data. Assigning transparency properties to the data, and using ray tracing is one possibility being discussed. Slicing the data volume with hyperplanes allows the use of bivariate rendering routines directly. The problem of approximating and modeling contours of scalar fields is specifically emphasized.

The common approach treating scalar valued data in space consists of a two step process. An approximating function to the given data is computed, later typically rendered using contour plots. A different sequence of modeling steps is proposed here. First, a piecewise linear approximation to a contour is constructed from the given data yielding a set of triangulated surfaces. Second, all triangulated surfaces

are used for generating smooth contours. Scalar fields can be discontinuous by nature, therefore determining a data set rapidly changing within small distances. The boundaries of internal structures in a volume might be given by a contour level close to a discontinuity. Computerized axial tomography (CAT) is an example for the advantage of the new method. Scanning a bone yields thousands of density values which consequently makes an overall approximation quite expensive. Because it is the shape of the bone which is of interest, the contour corresponding to the bone's boundary should be considered only.

A technique for approximating curvatures for surfaces as well as for trivariate functions are inferred from differential geometry. Curvature approximation is important both as input for surface schemes and as a tool for smoothness analysis. A data reduction algorithm is introduced that iteratively eliminates knots for a general triangulated surface, and a new scheme for producing a tangent-plane-continuous surface is presented.

To those

I love and respect,

Hanna and Günter Hamann,

Peter Hackemann,

and

Christian Pfennig

Acknowledgements

I am particularly thankful for the help and guidance of my advisor Dr. Gregory M. Nielson. He assisted me in preparing and conducting research and in presenting it to the scientific community. I further express my gratitude to Dr. Robert E. Barnhill, Dr. Gerald Farin, Dr. Thomas A. Foley, and Dr. Alyn P. Rockwood who not only supported me during the process of creating this dissertation, but also taught me very different ways of describing and solving mathematical problems using diverse tools. They all had a major influence on me as a researcher. I also appreciate the help of all other members of the CAGD research group at Arizona State University and all the benefits I gained in the many discussions I had with them. In addition, I thank Wayne B. Woodland for his assistance in composing most of the pictures and Joseph A. Reuter for his help. This research was supported by the Department of Energy under contract DE-FG02-87ER25041 and by the National Science Foundation under contract DDM 8807747.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Computerized axial tomography (CAT) and magnetic resonance imaging (MRI) have been major breakthroughs in medical imaging. Both rendering techniques supply people in the field of medical diagnosis with two-dimensional images revealing internal structures of some part of the human body. They are based on slicing a three-dimensional object and producing a sequence of two-dimensional pictures. Scanning a human's head, CAT produces slices taken perpendicular to the spine. Each slice delivers an X-ray image representing density values in a particular plane. The pictures obtained by this method show great detail in bone structures (see Figure 1.1., created by the slicing method described in chapter 2.3.). MRI, on the other hand, allows slicing a head in axial (top-to-bottom), saggital (ear-to-ear), and coronal (nose-to-back) direction directly. It also gives better insight into soft tissue. Looking at these slices requires a person's ability to intuitively interpolate between them in order to get a perception of the real three-dimensional structure. People in the medical field are quite capable in performing this task, but creating three-dimensional images on a computer screen instantaneously might be an even more powerful tool in diagnosis.

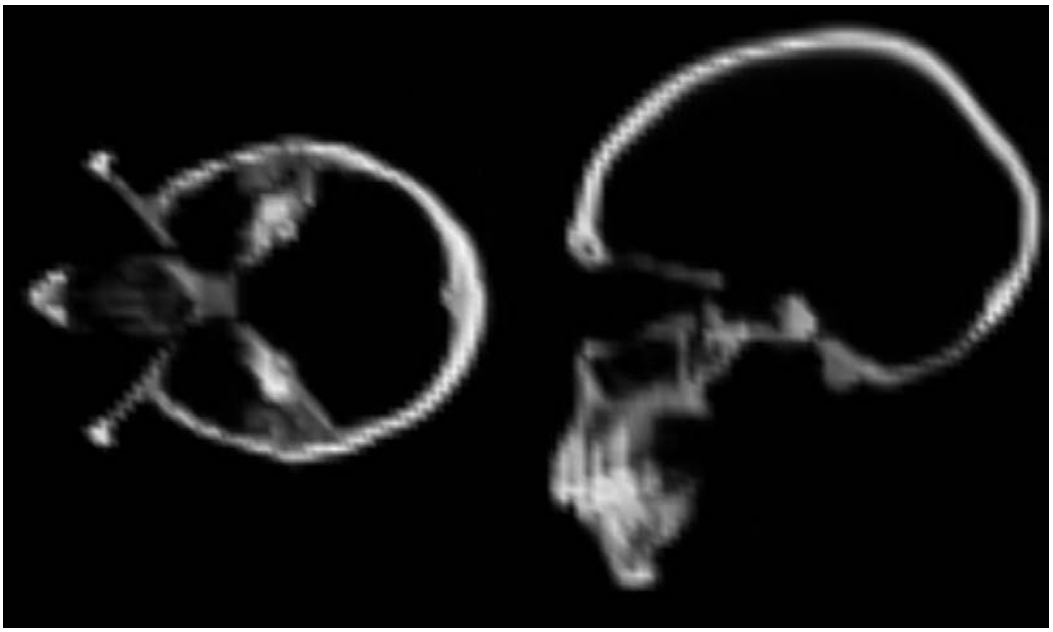

Fig. 1.1. Axial and saggital slice of human head (CAT).

CAT and MRI are examples serving well as a prelude for the topic of this dissertation. Visualizing and approximating three-dimensional data belonging to a scalar field (points in space with associated function values) is a difficult problem, too, in meteorology (temperature, pressure, velocity measurements), in geography (physical maps), physics, and mathematics itself. Mathematically, the data given can be viewed as the discretization of an unknown scalar field with three spatial variables sampled at certain points. It is quite common to approximate the data with a trivariate function over some domain (either inside the bounding box or the convex hull of the data points), evaluate the trivariate function on a regular grid and then plot the result. New methods are developed for both visualizing and modeling these data.

Several approaches exist for visualizing bivariate functions. Among the most common ones are contour lines (often called isolines) in the plane and surfaces in three-dimensional space. Using the second possibility, a point on the surface is simply given by the two spatial variables in the plane and the function value there (which yields the perpendicular distance to the plane). These techniques are extended to the trivariate case. Subdividing the three-dimensional domain of a trivariate function, simulating transparency for volumetric data, and slicing the domain with hyperplanes are methods being investigated.

A new path is followed modeling trivariate data. Traditionally, a continuous-approximating function is constructed considering each of the given data points and then evaluated. From a mathematical point of view, this is an obvious path to choose. However, some physical data sets might belong to a discontinuous scalar field. Discontinuities might have their explanation in the fact that physically different objects are present within the data. Different objects with their different properties naturally lead to rather heterogeneous function values (measurements) associated with them. Referring to the previous example of CAT and MRI, thousands of density measurements are obtained throughout a volume. In those applications, the internal structures in the volume and their actual three-dimensional geometry are of much greater interest than a trivariate function approximating all the thousands of density values.

Algorithms producing precisely these internal structures (contours of the density function) are reviewed and extended such that a topologically complete rep-

resentation of the boundaries of different objects is obtained. Being surfaces in three-dimensional space, these boundaries are described in terms of triangulations. Each triangulation belonging to a particular object (a single component of a contour) can now be treated separately.

The triangulations in space might still be redundant in the sense that too many triangles might be used to approximate the shape of contours. A data reduction algorithm is developed examining the curvature of a surface. The number of triangles in nearly planar regions is reduced significantly. The method used for locally approximating the curvature of a polygonized surface in three-dimensional space generalizes nicely to the case of a trivariate function whose domain is subdivided into a set of tetrahedra. Curvature, in this generalized case, becomes more complicated, but it still helps to reveal qualitative properties of trivariate functions.

Finally, the set of reduced triangulations approximating different components of a contour is used to construct tangent-plane-continuous surfaces. A simple rational curve scheme based on degree elevated conics is used for this purpose. Briefly, methods for estimating normal vectors, needed for this construction, are discussed.

Overall, the sequence of modeling steps proposed in this dissertation makes use of intrinsic properties of the data. If the existence of different objects within a three-dimensional volume is known, and the corresponding contour level is apparent, this new strategy is more satisfactory with respect to both storage and computing efficiency.

Chapter 2 shows a variety of rendering methods for trivariate data. In chapter

3, an algorithm is described for generating a piecewise linear approximation of a contour. The result of this algorithm is a set of contour points, representing the geometry of the contour in three-dimensional space, and topological information. The topological information relates contour points to each other, determining which points form the vertices in a triangulation; it also includes the neighborhood information in the triangulation and associates each triangle with the component of a contour it belongs to. The chapter is concluded with some remarks concerning the estimation of gradients and normal vectors for three-dimensional points.

In chapter 4, principal curvatures are approximated for both surfaces and trivariate functions (precisely, for two-dimensional and three-dimensional manifolds). The concept of an osculating paraboloid (as used in differential geometry) derived from a local least squares fit serves as mathematical tool for this purpose. Chapter 5 presents a data reduction technique for a triangulated surface. Triangles are iteratively removed in areas with low curvature. In chapter 6, the reduced triangulation is used to construct an overall tangent-plane-continuous surface. Chapter 7 concludes the dissertation, summarizing the main results and mentioning possibilities for further research.

All algorithms described in this dissertation have been implemented in the programming language C on a Silicon Graphics workstation, model 4D/220 GTX.

Chapter 2

Visualization techniques for trivariate data

2.1. Existing methods and terminology

Several methodologies for visualizing trivariate data have been outlined in the first chapter. Contemporary computer graphics equipment is capable of performing millions of arithmetic operations per second, providing an excellent tool for rendering three-dimensional data sets in real time. All techniques presented here are designed to allow interaction with the user.

Imaging three-dimensional data is known as **volume visualization** or **volume rendering**. Algorithms based on ray-tracing have been widely used for volumetric data, e.g., in the medical field ([Fuchs et al. '89], [Kajiya & Herzen '84], [Levoy '88,'90], [Ney et al. '90], [Sabella '88], [Tiede et al. '90]). Considering the huge amount of data, ray-tracing is undoubtedly computing-intensive and hardly interactive. In [Foley et al. '90] nearly real time computing of ray-traced images is discussed. All these algorithms simulate the behavior of light rays (X-rays, for instance) passing through a finite volume containing the data to be visualized. "Objects" inside this volume usually appear more or less translucent. In Figure 2.1., CAT scan data (68 axial slices, each containing $64 \cdot 64$ integer density values in $\{0, 1, ..., 255\}$) of a human skull are rendered using the algorithm from [Levoy '88].

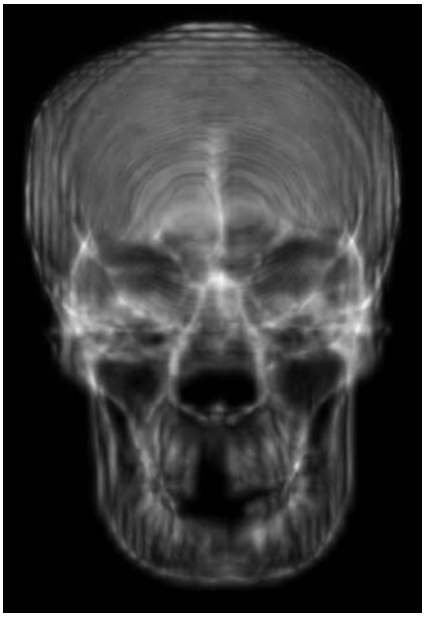

Fig.2.1. Skull rendered using Levoy's algorithm for CAT scan data.

Several rendering techniques are extensions of the bivariate to the trivariate case. In order to make use of lower-dimensional methods, a three-dimensional volume is intersected with a hyperplane, the trivariate function is restricted to this plane and rendered only for the single slice obtained (see examples in [Banchoff '90]). Numerous volume visualization techniques based on drawing graphs of bivariate functions and general overviews can be found in [Drebin et al. '88], [Nielson et al. '91], [Hamann '90a], and [Nielson & Hamann '90]. Algorithms based on approximating contours of trivariate functions are not discussed in this chapter. Generating linear contour approximations and modeling them is the content of the following chapters. A definition is given to understand the common nomenclature used in combination with visualizing and modeling multivariate data.

Definition 2.1. A **scattered trivariate data set** is the set

$$\left\{ \ ({\mathbf{x}_i}^T, f_i) = (x_i, y_i, z_i, f_i) \mid \mathbf{x}_i \in I\!R^3, \ f_i \in I\!R, \ i = 0...n \ \right\}, \tag{2.1.}$$

a **rectilinear trivariate data set** is the set

$$\left\{ \ ({\mathbf{x_i}}^T, f_\mathbf{i}) = (x_i, y_j, z_k, f_{i,j,k}) \mid \mathbf{x_i} \in I\!R^3, \ f_\mathbf{i} \in I\!R, \ i = 0...n_x, j = 0...n_y, k = 0...n_z \ \right\}. \tag{2.2.}$$

Often, these two data sets are simply referred to as scattered or rectilinear data. It is this kind of data that is visualized and modelled. Physical measurements are usually given as scattered data, whereas rectilinear data arise from evaluating some known trivariate function in a structured fashion. The geometry of an equidistantly-spaced rectilinear data set is directly reflected by the data's indices:

$$\begin{aligned} x_i &= x_0 + i \ \frac{x_{n_x} - x_0}{n_x}, \qquad i = 0...n_x, \\ y_j &= y_0 + j \ \frac{y_{n_y} - y_0}{n_y}, \qquad j = 0...n_y, \\ z_k &= z_0 + k \ \frac{z_{n_z} - z_0}{n_z}, \qquad k = 0...n_z. \end{aligned}$$

The convex hull C of a rectilinear data set is therefore given by

$$C = [x_0, x_{n_x}] \times [y_0, y_{n_y}] \times [z_0, z_{n_z}] = [x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}].$$

In the case of scattered data, there is no underlying structure hidden in the geometry of the data points. For modeling purposes, the convex hull of scattered data points is split into a set of tetrahedra, usually, to obtain the Delaunay triangulation implied by the points (see [Preparata & Shamos '90]).

The different approaches used in computer-aided geometric design to model scattered or rectilinear data are not reviewed in detail in this dissertation. Generally, the modeling process can be divided into derivative estimation (typically, first and second order derivatives), construction of some trivariate function approximating the given function values and evaluation and visualization of that function. Normally, the function constructed to approximate either scattered or rectilinear data is evaluated on a rectilinear grid. This is the motive to primarily develop rendering techniques for rectilinear data sets.

Methods addressing the problem of estimating derivatives and approximating scattered and rectilinear data can be found in [Alfeld '89], [Barnhill & Little '84], [Boehm et al. '84], [Dahmen '89], [Davis '75], [Farin '83], [Farin '90], [Franke & Nielson '91], [Hoschek & Lasser '89], [Sederberg '85], [Stead '84], and [Worsey & Farin '87].

2.2. Domain subdivision and transparency techniques

The objective is to develop simple algorithms which can easily be implemented and can be used in a real-time, interactive fashion. All algorithms described here visualize rectilinear data sets. The domain-subdivision technique is based on extracting subvolumes $V_{r,s,t}$ from the convex hull C of the data points and coloring the surfaces of these subvolumes according to the function values on the surfaces of each such subvolume ($V_{r,s,t}$ is a box defined by its width, depth, and height).

Here, the idea is to allow for space between all subvolumes so that it is possible

to "look inside" the data set when all subvolumes are rendered simultaneously. It is sufficient for this technique to specify resolution parameters q_x, q_y, and q_z for the three spatial directions and ratios α_x, α_y, and α_z determining the relative length of space between two consecutive subvolumes and an edge of a subvolume. Denoting the three edge lengths of a subvolume by $\triangle x$, $\triangle y$, and $\triangle z$, a subvolume $V_{r,s,t}$ is given by its left-front-lower corner point $(x_r, y_s, z_t)^T$ and its three edge lengths:

$$x_r = x_{min} + r\ (1+\alpha_x)\ \triangle x, \qquad \triangle x = \frac{x_{max} - x_{min}}{q_x + \alpha_x(q_x - 1)}, \qquad r = 0...(q_x - 1),$$

$$y_s = y_{min} + s\ (1+\alpha_y)\ \triangle y, \qquad \triangle y = \frac{y_{max} - y_{min}}{q_y + \alpha_y(q_y - 1)}, \qquad s = 0...(q_y - 1),$$

$$z_t = z_{min} + t\ (1+\alpha_z)\ \triangle z, \qquad \triangle z = \frac{z_{max} - z_{min}}{q_z + \alpha_z(q_z - 1)}, \qquad t = 0...(q_z - 1).$$

If the function to be rendered is known, it is evaluated at the eight corner points of each subvolume $V_{r,s,t}$; if it is not known, i.e., one is solely given a rectilinear data set, function values for $V_{r,s,t}$'s corner points are obtained by trilinear interpolation of those eight given function values in the rectilinear data set associated with a certain corner point. For example, if $x_r \in [x_i, x_{i+1}]$, $y_s \in [y_j, y_{j+1}]$, and $z_t \in [z_k, z_{k+1}]$, the function value at $(x_r, y_s, z_t)^T$ is the value of the trilinear interpolant to the set of known values $\{f_{i,j,k}, f_{i+1,j,k}, ..., f_{i+1,j+1,k+1}\}$ at $(x_r, y_s, z_t)^T$.

Now, minimal and maximal function values are determined among the corner function values of all subvolumes. They are denoted by f_{min} and f_{max}. A linear map is used to assign (integer) color values to each corner point of each subvolume. If c_{min} and c_{max} are the extreme (integer) color indices referring to a predefined

color map, a (real) corner function value f is mapped to the color (-index) c, where

$$c \;=\; \left\lceil \frac{f_{max}-f}{f_{max}-f_{min}}\; c_{min} + \frac{f-f_{min}}{f_{max}-f_{min}}\; c_{max} \right\rceil .$$

Each face of the subvolumes is then Gouraud-shaded, i.e., a face's color is obtained by bilinear interpolation of the four colors (color-indices) associated with that face. If the function to be rendered is known and the extension of a subvolume $V_{r,s,t}$ (given by $\triangle x$, $\triangle y$, $\triangle z$) is relatively large compared to the extension of the convex hull C of the entire data set, it is appropriate to evaluate the function at more points than at the eight corner points of each subvolume. In principle, $(k+1)(l+1)$ function (and color) values are determined for points $\mathbf{x}_{i,j}, i = 0...k, j = 0...l$, arranged in a rectilinear fashion on a subvolume's face. Each two-dimensional grid cell on a single face, given by the four points $\mathbf{x}_{i,j}$, $\mathbf{x}_{i+1,j}$, $\mathbf{x}_{i,j+1}$, and $\mathbf{x}_{i+1,j+1}$, is then Gouraud-shaded itself.

Rendering all $q_x q_y q_z$ subvolumes and rotating them in real-time is possible. The impression of the three-dimensional structure can further be improved by drawing lines between the centroid of $V_{r,s,t}$ and the centroids of the six "neighbor" subvolumes $V_{r-1,s,t}$, $V_{r+1,s,t}$, ..., and $V_{r,s,t+1}$. Two examples for the domain-subdivision method using different resolution parameters are shown in Figures 2.2. and 2.3. The trivariate function visualized in both cases is

$$f\ (x,y,z) = 15\ (\quad e^{-0.005((x-10)^2+(y-10)^2+(z-10)^2)}$$

$$+\, e^{-0.0025((x-15)^2+(y-20)^2+(z-20)^2)} + e^{-0.005((x-25)^2+(y-25)^2+(z-25)^2)}),$$

$x, y, z \in [0, 39]$. Low values are mapped to green, high ones to white.

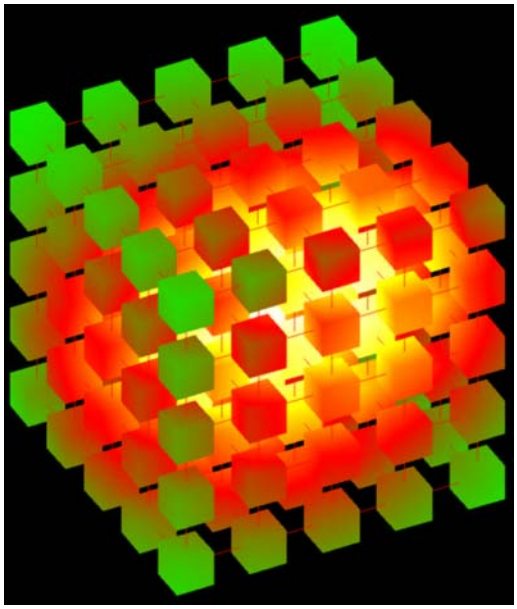

Fig. 2.2. Domain subdivision method for exponential function, $q_x = q_y = q_z = 5, \quad \alpha_x = \alpha_y = \alpha_z = 1.$

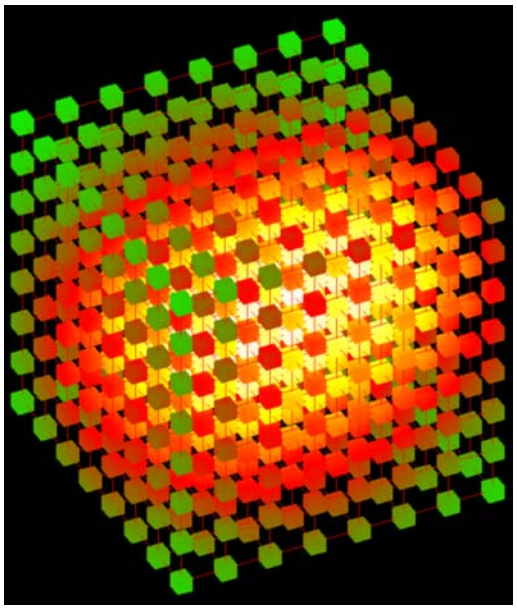

Fig. 2.3. Domain subdivision method for exponential function, $q_x = q_y = q_z = 8, \quad \alpha_x = \alpha_y = \alpha_z = 2.$

The domain-subdivision technique can also be applied to volumes obtained by a map from three-dimensional $uvw-$space to three-dimensional $xyz-$space,

$$(x, y, z) = \big(\ x(u, v, w), \ y(u, v, w), \ z(u, v, w) \ \big),$$

$$u \in [u_{min}, u_{max}], \ v \in [v_{min}, v_{max}], \ w \in [w_{min}, w_{max}].$$

Here, x, y, and z are continuous functions in all three variables. The function to be rendered is defined in $xyz-$space and the subdivision parameters actually refer to $uvw-$space. Using this approach, it is possible to visualize functions defined over more general volumes, such as volumes bounded by spheres, ellipsoids or paraboloids. Figure 2.4. shows the function

$$f \ (x, y, z) = \cos \ \big(\ 2 \ \sqrt{x^2 + y^2 + z^2} \ \big),$$

defined over a part of the unit ball.

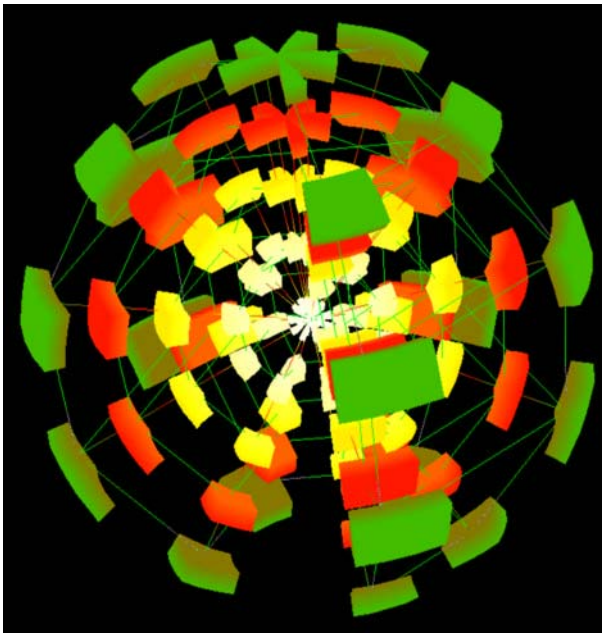

Fig. 2.4. Domain subdivision method for trigonometric function, unit ball, $q_u = q_v = q_w = 5, \quad \alpha_u = \alpha_v = \alpha_w = 2.$

The volume considered is the set of points $(x, y, z)^T$, where

$$(x, y, z) = (\ u\cos v,\ u\sin v\cos w,\ u\sin v\sin w\ ),\ u \in [0, 1],\ \ v \in [0, \pi],\ \ w \in [0, \tfrac{3}{2}\pi].$$

Some graphics-workstations support a technique called *alpha blending*. It is a tool used for rendering transparent objects. Objects to be visualized in three-dimensional space are approximated by a set of planar polygons, where the term *polygon* is understood as the area bounded by a planar, closed, and non-self - intersecting piecewise linear curve. Alpha-blending requires an ordered set of polygons sorted with respect to their distance to the screen. Among a polygon's corner points one can use the one closest to the screen as sorting criterion.

A parameter, denoted by t, specifies the degree of transparency of polygons. If c_{old} is the color currently displayed at a screen position (i, j), and another polygon is to be rendered covering this particular area of the screen, the new color c_{new} for (i, j) is obtained by linear interpolation of the current color and c_{poly}, the color of the polygon at $(i, j) : c_{new} = (1 - t)c_{poly} + tc_{old}$, $t \in [0, 1]$. This implies that objects appear non-transparent, if t is 0.

In order to utilize alpha-blending, a trivariate function defined over $[x_0, x_{n_x}] \times [y_0, y_{n_y}] \times [z_0, z_{n_z}]$ is evaluated on a rectilinear grid yielding $(n_x + 1)(n_y + 1)(n_z + 1)$ points and function values. Three sets of polygons are constructed: the first set consists of rectangles parallel to the $xy-$plane, the second of rectangles parallel to the $xz-$plane, and the third of rectangles parallel to the $yz-$plane. Each rectangle in the rectilinear grid in the first set is defined by the points $\mathbf{x}_{i,j,k}$, $\mathbf{x}_{i+1,j,k}$, $\mathbf{x}_{i,j+1,k}$, and $\mathbf{x}_{i+1,j+1,k}$, $i = 0...(n_x - 1), j = 0...(n_y - 1), k = 0...n_z$. Similarly, one generates

the rectangles of the other two sets. Again, the function values at these points determine the colors used for rendering.

The transparency parameter t and an orientation for the domain must be specified. Sorting all 3 $n_x n_y n_z + n_x n_y + n_x n_z + n_y n_z$ rectangles becomes an obstacle for using alpha-blending as an interactive visualization technique with high resolution parameters n_x, n_y, and n_z. Real-time performance can be achieved for varying t and fixed domain-orientation, but not conversely. The re-sorting of all rectangles takes too long in this case. In Figures 2.5. and 2.6., a gas-concentration is shown using different values for t (see [Long et al. '89]). Transparency definitely increases the visual understanding of a trivariate function by providing the possibility to perceive contours inside the function's domain.

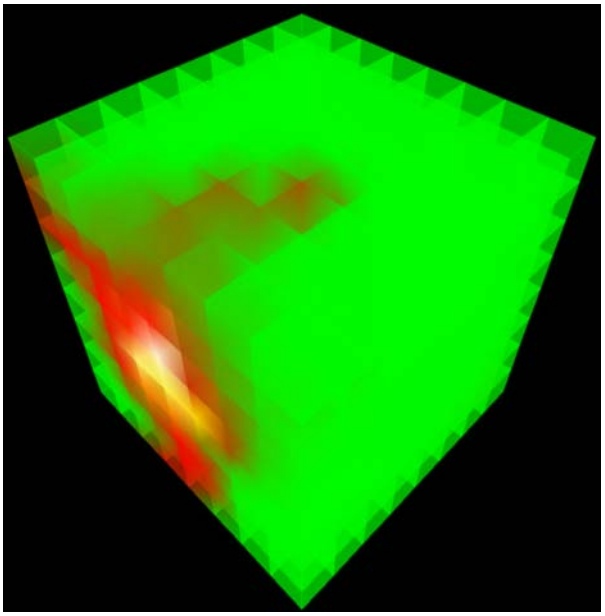

Fig. 2.5. Domain subdivision and transparency,
$n_x = n_y = n_z = 8, \quad t = 0.5.$

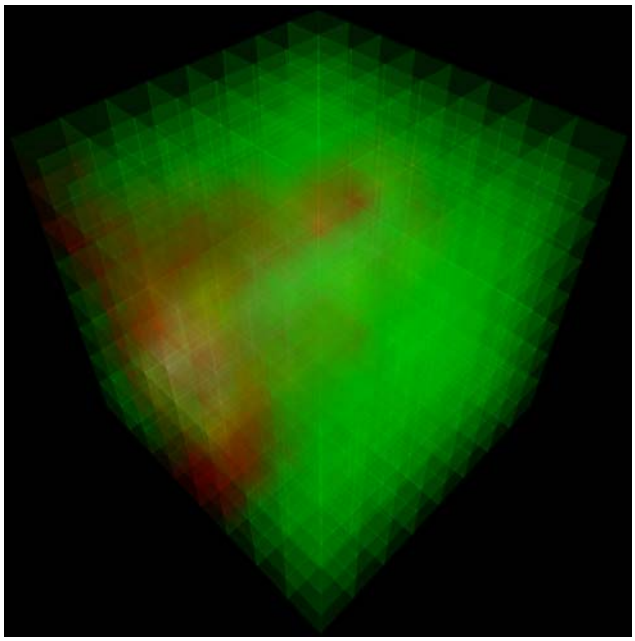

Fig. 2.6. Domain subdivision and transparency,
$n_x = n_y = n_z = 8, \quad t = 0.95.$

2.3. Slicing methods

Slicing methods are based on intersecting the domain D of a trivariate function with a hyperplane

$$P = \left\{ \mathbf{x} \mid (\mathbf{x} - \mathbf{x}_0) \cdot \mathbf{n} = 0, \ \mathbf{x}, \mathbf{x}_0 \in I\!R^3, \ \mathbf{n} \ \text{ normal to } \ P \right\}$$

usually determining an area A bounded by a closed polygon. The trivariate function to be rendered is then restricted to A, evaluated and rendered for A only.

It is supposed that D is $[x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}]$, and the hyperplanes used have normal vectors $\mathbf{n}_1 = (1,0,0)^T$, $\mathbf{n}_2 = (0,1,0)^T$, and $\mathbf{n}_3 = (0,0,1)^T$. Three mutually perpendicular planes $P_i, i = 1,2,3$, are defined such that

$A_i = P_i \cap D \neq \emptyset$. The trivariate function is then evaluated on three $(k_i + 1)(l_i + 1)$ rectilinear grids, one grid per area A_i. These three areas can be visualized in two different ways.

The first possibility to visualize the function on all three areas A_i can be characterized as follows:

- assign colors (color indices referring to a predefined color map)
  to each point in each rectilinear grid,
- use Gouraud shading to color each rectangle given by the points
  $\mathbf{x}^i_{r,s}$, $\mathbf{x}^i_{r+1,s}$, $\mathbf{x}^i_{r,s+1}$, $\mathbf{x}^i_{r+1,s+1}$, $r = 0...(k_i - 1)$, $s = 0...(l_i - 1)$,
  $i = 1, 2, 3$, in each rectilinear grid and
- allow the user to interactively move any of the hyperplanes P_i
  in $x-$, $y-$, and $z-$direction, respectively.

Rotating the three hyperplanes in real-time and modifying the resolution parameters k_i and l_i can be accomplished interactively as well. In Figures 2.7. and 2.8., the same gas-concentration is visualized as in Figures 2.5. and 2.6. Depending on the data to be rendered, one might prefer a color map using multiple colors or a single color.

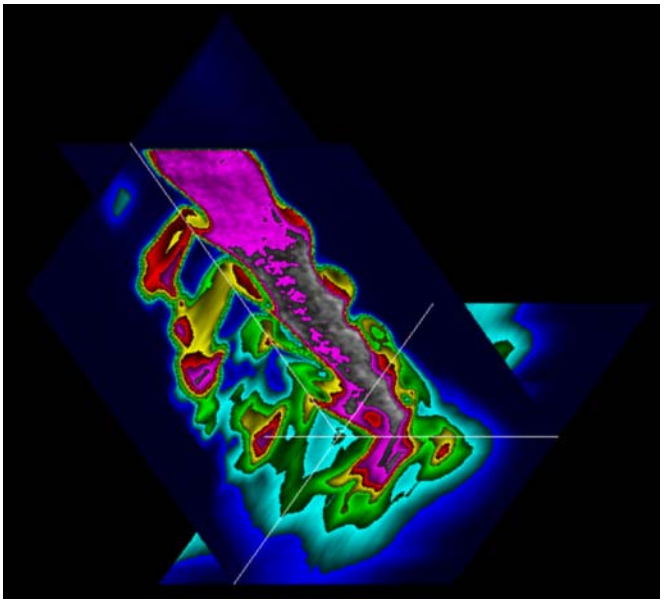

Fig. 2.7. Slicing method, coloring hyperplanes, multiple colors, $k_1 = l_1 = 80,\ k_2 = l_2 = 130,\ k_3 = l_3 = 20.$

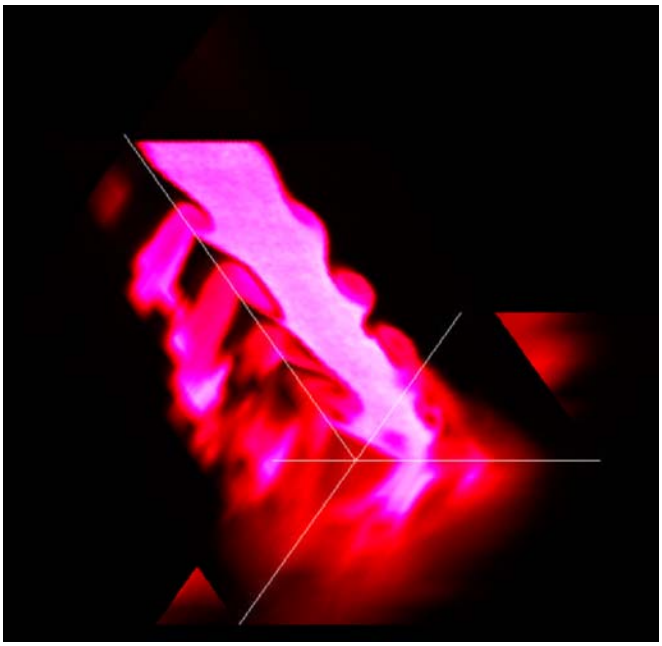

Fig. 2.8. Slicing method, coloring hyperplanes, single color, $k_1 = l_1 = 80,\ k_2 = l_2 = 130,\ k_3 = l_3 = 20.$

The second possibility to visualize the function on all three areas A_i is similar to the standard procedure rendering a bivariate function. One generally evaluates a bivariate function over a specified (rectangular) domain. The result is a set of two-dimensional (domain-) points with associated function values. Points and function values combined are then interpreted as a set X of three-dimensional points yielding the graph of the bivariate function:

$$X \;=\; \{\; (x_i, y_j, f_{i,j})^T \mid i = 0...k,\; j = 0...l \;\}.$$

The points in X are usually mapped into $[0,1]^3$. The graph can either be a curve network of piecewise linear curves or a shaded surface. In the first case, curves are defined by the line segments

$$\overline{(x_i, y_j, f_{i,j})^T (x_{i+1}, y_j, f_{i+1,j})^T}, \qquad i = 0...(k-1),\; j = 0...l \qquad \text{and}$$

$$\overline{(x_i, y_j, f_{i,j})^T (x_i, y_{j+1}, f_{i,j+1})^T}, \qquad i = 0...k,\; j = 0...(l-1).$$

In the second case, a surface is defined by two sets, I_1 and I_2, of index triples, each triple referring to three points in X,

$$I_1 \;=\; \Big\{\; \big((i,j),(i+1,j),(i+1,j+1)\big) \mid i = 0...(k-1),\; j = 0...(l-1) \;\Big\} \qquad \text{and}$$

$$I_2 \;=\; \Big\{\; \big((i,j),(i+1,j+1),(i,j+1)\big) \mid i = 0...(k-1),\; j = 0...(l-1) \;\Big\}.$$

Each triangle determined by an index triple is shaded on the screen.

To utilize this rendering technique for a trivariate function, the function is restricted to the three areas A_i, $i = 1, 2, 3$, again, evaluated on a rectilinear grid for each A_i and finally visualized as a set of three graphs, each graph either a curve network or a shaded surface. It is convenient to choose $P_1 = \{\mathbf{x} | x = c_1 \in [x_{min}, x_{max}]\}$, $P_2 = \{\mathbf{x} | y = c_2 \in [y_{min}, y_{max}]\}$ and $P_3 = \{\mathbf{x} | z = c_3 \in [z_{min}, z_{max}]\}$.

The curve networks or shaded surfaces are then plotted over planes parallel to the $xy-$, $xz-$, and $yz-$plane, respectively. Special care must be taken to avoid intersections among the three graphs. To gain a better visual understanding of the position and orientation of the three areas A_i relative to each other and to the trivariate function's domain, the areas A_i are represented as single-colored rectangles in a box indicating the function's domain. Figure 2.9. is an example for this surface-based rendering technique using the same exponential function as in Figures 2.2. and 2.3.

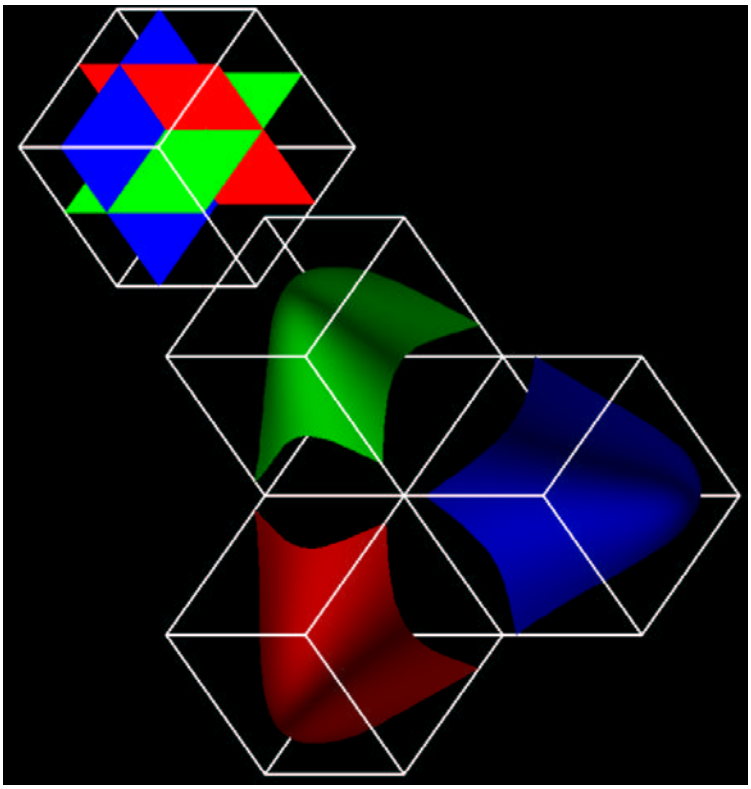

Fig. 2.9. Slicing method, bivariate surfaces, Gouraud-shaded, $k_1 = k_2 = k_3 \ = \ l_1 = l_2 = l_3 \ = \ 30$.

Chapter 3

Trilinear contour approximation for trivariate data

3.1. Previous work and basic definitions

Visualizing contour levels of a trivariate function is another possibility to obtain insight into a function's behavior by displaying them as surfaces. This has not yet been examined in greater detail in chapter 2, due to the fact that contours are not primarily approximated for rendering purposes, but for modeling the contours themselves as surfaces in three-dimensional space.

In principle, a point set is determined such that each point in this set is close to a certain contour, the point set is triangulated (yielding a two-dimensional triangulation in three-dimensional space), the neighbors for each triangle in the resulting triangulation are determined, and each triangle is associated with a particular part of the contour. Normal vectors are estimated for each point, needed for further modeling the data.

In [Bloomquist '90], [Petersen '84], and [Petersen et al. '87] different approaches are described to contour trivariate functions given in explicit form. Other references can be found there as well. Approximating contours from rectilinear trivariate data sets alone is explored in [Hamann '90b], [Lorensen & Cline '87], and [Nielson & Hamann '91b]. An error in the *marching-cubes method* by Lorensen has been pointed out in [Dürst '88]: Approximating a contour using Lorensen's technique results in a triangulation which might lead to "holes" (locally missing or improperly constructed triangles) for special data configurations. An optimization algorithm

for a two-dimensional triangulation in three-dimensional space is given in [Choi et al. '88].

Ways for resolving the inaccuracy in Lorensen's contouring method are shown in 3.2. For further modeling this piecewise planar contour approximation, derivative information must be provided. Estimating gradients for trivariate functions is discussed in [Stead '84] and [Zucker & Hummel '81]. These estimates determining additional geometrical information (tangent planes with orientation) for the vertices in the triangulation are needed to create overall tangent-plane-continuous surfaces for each part of a contour. Again, contours of trivariate functions are interpreted here as two-dimensional boundaries of objects. Therefore, triangulations approximating such contours are used as input for a surface scheme.

Definition 3.1. Let $f : \mathbb{R}^3 \rightarrow \mathbb{R}$; a trivariate **contour** is the point set

$$C_f(\alpha) \;=\; \left\{\; \mathbf{x} \mid f(\mathbf{x}) = \alpha,\; \alpha \in \mathbb{R} \;\right\} \;\subset\; \mathbb{R}^3. \tag{3.1.}$$

Contours of trivariate functions are also referred to as **contour surfaces**, **isosurfaces**, **level surfaces** or **niveau sets**.

A contour might be partitioned into several unconnected subsets. This motivates the next definition.

Definition 3.2. Let $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ be a continuous function; a **component** $C_f^j(\alpha) \subseteq C_f(\alpha)$ is the set of points such that for each pair of points $\mathbf{x}, \mathbf{y} \in C_f^j(\alpha)$ a curve $\mathbf{c} \subset C_f^j(\alpha)$ exists connecting $\mathbf{x}$ and $\mathbf{y}$.

Definition 3.3. Let $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ be a partially differentiable function (a C^1 function). The **gradient** of f in $\mathbf{x} \in \mathbb{R}^3$ is the triple

$$\bigtriangledown f(\mathbf{x}) = \Big(\frac{\partial f}{\partial x}(\mathbf{x}),\ \frac{\partial f}{\partial y}(\mathbf{x}),\ \frac{\partial f}{\partial z}(\mathbf{x}) \Big) = \big(f_x(\mathbf{x}),\ f_y(\mathbf{x}),\ f_z(\mathbf{x}) \big). \tag{3.2.}$$

Theorem 3.1. *Let $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ be a C^1 function and $\bigtriangledown f(\mathbf{x}) \neq (0,0,0)$ for all $\mathbf{x} \in \mathbb{R}^3$. Then, every contour $C_f(\alpha)$ is a surface (a two-dimensional manifold).*

Proof. The Taylor series for f at a point $\mathbf{x}_0 \in C_f(\alpha)$ is

$$f(\mathbf{x}) = \bigtriangledown f(\mathbf{x}_0)\ (x - x_0, y - y_0, z - z_0)^T + R(\mathbf{x}) = 0.$$

This is the equation of an implicitly defined surface of at least first degree.

q.e.d.

Remark 3.1. For computing purposes it must be assured that a contour of a trivariate function is not (locally) a three-dimensional volume. This would be the case if $f = \alpha$ on such a volume, implying a vanishing gradient.

For the further discussion the domain of the triavariate function is restricted to a subset of $\mathbb{R}^3$. In most applications this subset is a box. This restriction implies that a single component of a contour can be divided into several unconnected parts inside a subset. Therefore, the following definitions are necessary.

Definition 3.4. Let $U = (x_0, x_1) \times (y_0, y_1) \times (z_0, z_1) \subset \mathbb{R}^3$ and $\overline{U}$ be the closure of U. If

$$B \;=\; (\overline{U} \setminus U) \cap C_f(\alpha) \;\neq\; \emptyset,$$

then B is called the **boundary** of $C_f(\alpha)$ with respect to U.

Remark 3.2. $\overline{U} \setminus U$ constitutes the faces of $\overline{U}$.

Definition 3.5. Let U be defined as in 3.4. and $C_f(\alpha) = \bigcup_{j=1}^{m} C_f^j(\alpha)$ be a contour of a function $f : I\!R^3 \rightarrow I\!R$; a **part**

$$P_j^k|_{\overline{U}},\ j = 1...m,\ k = 1...m_j, \text{ of a component } C_f^j(\alpha),\ P_j^k(\overline{U}) \subseteq C_f^j(\alpha),$$

is the subset of points such that for each pair of points $\mathbf{x}, \mathbf{y} \in P_j^k|_{\overline{U}}$ a curve $\mathbf{c} \subset P_j^k|_{\overline{U}}$ exists connecting $\mathbf{x}$ and $\mathbf{y}$.

Remark 3.3. If a part $P_j^k|_{\overline{U}}$ of a component $C_f^j(\alpha)$ coincides with a face, an edge or a corner of $\overline{U}$, a special case treatment is necessary. Only for the first of these three cases a part is considered, in the other two cases the dimension of the part (relative to $\overline{U}$) is less than 2 and therefore neglected.

Remark 3.4. It is usually difficult to determine that two parts belong to the same component of a contour, if one restricts oneself to $\overline{U}$. As a result of this, the term part is used only, the connection between parts and the component they actually belong to is no longer made when limiting a contour to $\overline{U}$.

A contour of an arbitrary trivariate function usually can not be described in an explicit form. For this reason, a finite set of points is created, each point lying on the contour. This point set is then triangulated to yield a piecewise planar approximation to the true contour.

Definition 3.6. Let $P_j^k|_{\overline{U}}$ be a part of a component such that it has a non-empty intersection with the faces of $\overline{U}$, $P_j^k|_{\overline{U}} \cap (\overline{U} \setminus U) \neq \emptyset$. Let Y be a finite set of points

in $P_j^k(\overline{U})$, $Y = \{\mathbf{y} | \mathbf{y} \in P_j^k|_{\overline{U}}\}$. Y is a **closed contour point set** with respect to U, if it contains points $\mathbf{y}_r$ which can be ordered such that they describe a closed polygon on the faces of U,

$$\left\{ \overline{\mathbf{y}_r\mathbf{y}_{r+1}} \mid \overline{\mathbf{y}_r\mathbf{y}_{r+1}} \cap U = \emptyset, \mathbf{y}_r \in Y, r = 0...(p_m - 1), \text{ indices mod } p_m \right\}$$

is a closed polygon.

The segments defining such a polygon are called **boundary edges**.

Definition 3.7. Two triangles T_1 and T_2 are **neighbors**, if they have exactly two vertices in common.

The above definitions allow to introduce the term of a contour triangulation.

Definition 3.8. Let $C_f(\alpha) = \bigcup_{j=1}^{m} C_f^j(\alpha)$, $C_f^j(\alpha) = \bigcup_{k=1}^{m_j} P_j^k(\overline{U})$ be the contour of $f : I\!R^3 \rightarrow I\!R$ restricted onto $\overline{U}$, f being a C^1 function, such that f's gradient does not vanish on $C_f(\alpha)$, $\bigtriangledown f(\mathbf{x})\big|_{C_f(\alpha)} \neq (0,0,0)$. Let

$$X = \left\{ \mathbf{x}_i \mid \mathbf{x}_i \in C_f(\alpha), \ i = 1...n \right\} \subset \overline{U}$$

be a finite set of points in $C_f(\alpha)$. A **two-dimensional contour triangulation** $\mathcal{T}$ of the point set X is the set of index triples

$$\mathcal{T} = \left\{ T_j = (r_j, s_j, t_j) \mid r_j, s_j, t_j \in \{1, ..., n\}, \ r_j \neq s_j, \ r_j \neq t_j, \ s_j \neq t_j \right\} \quad (3.3.)$$

such that

(i) $\mathbf{x}_{r_j}$, $\mathbf{x}_{s_j}$, and $\mathbf{x}_{t_j}$ are the vertices of a triangle T_j,

(ii) each point in X is the vertex of at least one triangle,

(iii) the intersection of the interior of two triangles is empty,

(iv) an edge $\overline{\mathbf{xy}}$, $\mathbf{x}, \mathbf{y} \in X$, in the triangulation is shared by at most two triangles,

(v) each point $\mathbf{y}$ on a face of $\overline{U}$ belongs to exactly one closed contour point set Y, $\mathbf{y} \in Y \subset X$,

(vi) each triangle has exactly three neighbors, except those triangles having at least one boundary edge,

(vii) there is no edge connecting points $\mathbf{x}$ and $\mathbf{y}$, if $\mathbf{x} \in C_f^j(\alpha)$ and $\mathbf{y} \in C_f^k(\alpha)$, $j \neq k$, or, if $\mathbf{x} \in P_j^l|_{\overline{U}}$ and $\mathbf{y} \in P_j^m|_{\overline{U}}$, $l \neq m$,

(viii) T_j's outward unit normal $\mathbf{n}_j$ is defined as

$$\mathbf{n}_j = (\mathbf{x}_{s_j} - \mathbf{x}_{r_j}) \times (\mathbf{x}_{t_j} - \mathbf{x}_{r_j}) \;/\; ||(\mathbf{x}_{s_j} - \mathbf{x}_{r_j}) \times (\mathbf{x}_{t_j} - \mathbf{x}_{r_j})||,$$

where $||(x, y, z)^T|| = \sqrt{x^2 + y^2 + z^2}$.

Remark 3.5. There is no distinction made between the permutations of the index triples (r_j, s_j, t_j), (s_j, t_j, r_j), and (t_j, r_j, s_j); only the sequence of three indices in a triple determining a triangle's orientation is of importance ((viii) in Definition 3.8.). The term triangulation is used instead of two-dimensional contour triangulation whenever it is obvious from the context what is meant.

Figure 3.1. illustrates the concept of a triangulated contour divided into two parts inside a box (black dots at cell corners representing function values greater than the contour level α).

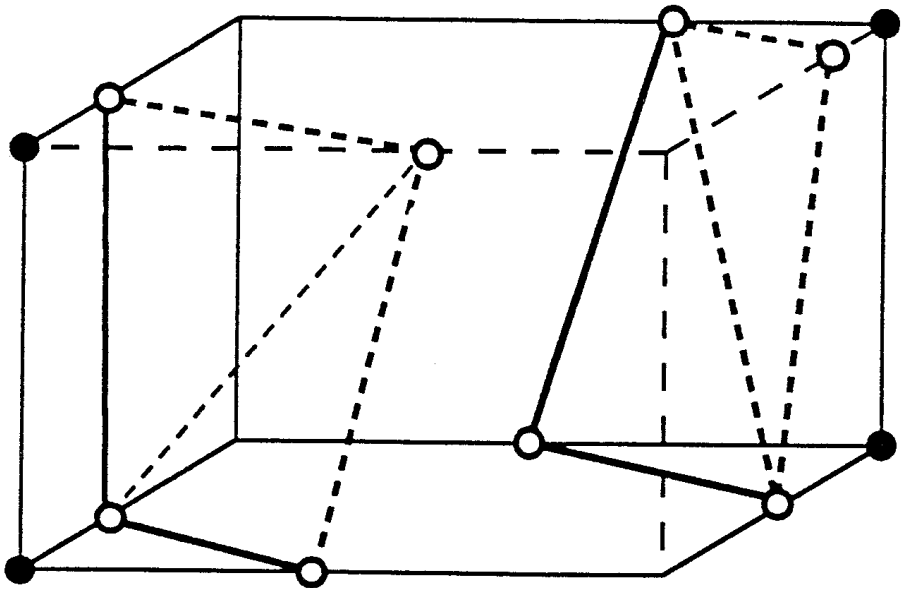

Fig. 3.1. Contour triangulation, contour divided into two parts.

To effectively test whether two triangles belong to the same part of a contour, a criterion must be given.

Definition 3.9. Two triangles $T_{l_1}, T_{l_m} \in \mathcal{T}$ **belong to the same part** $P_j^k|_{\overline{U}}$ of a component $C_f^j(\alpha)$ of a contour, if triangles $T_{l_2}, ..., T_{l_{m-1}} \in \mathcal{T}$ exist such that

$$\bigwedge_{i=1}^{m-1} \ (T_{l_i} \text{ and } T_{l_{i+1}} \text{ are neighbors }).$$

Definition 3.10. A **hole** in a contour triangulation $\mathcal{T}$ is defined by a set of m ordered edges

$$\left\{ \ e_i = \overline{\mathbf{x}_i \mathbf{x}_{i+1}} \mid i = 0...(m-1), \text{ indices } \mod \ m \ \right\}$$

forming a closed polygon, where each edge belongs to exactly one triangle in $\mathcal{T}$. A hole is an **interior hole** if at least one edge e_i is not a boundary edge.

Remark 3.6. Holes in a contour triangulation $\mathcal{T}$ can naturally occur because

the function f is restricted to $\overline{U}$. Interior holes are unnatural and undesired when approximating a trivariate contour with triangles in $\overline{U}$.

Definition 3.11. A two-dimensional contour triangulation $\mathcal{T}$ is **continuous**, if it does not contain interior holes.

3.2. Piecewise triangular contour approximation for rectilinear data

In several applications one is given a rectilinear data set as the result of physical measurements or simulations. Methods whose purpose is to approximate a contour of some underlying trivariate function (which is unknown itself) should take advantage of the structure implied by rectilinear data. An appropriate data element for a local contour approximation is a cell.

Definition 3.12. Let $X = \left\{(\mathbf{x_i}^T, f_\mathbf{i})\right\}$ be a rectilinear trivariate data set (Definition 2.1.). A **cell** $C_\mathbf{i}$ is the set of points

$$C_\mathbf{i} \;=\; [x_i, x_{i+1}] \times [y_j, y_{j+1}] \times [z_k, z_{k+1}],\ \mathbf{x_i} \in X.$$

Remark 3.7. The fact that three edges intersecting at a corner of a cell are mutually orthogonal to each other inspires the term rectilinear.

Remark 3.8. It is assumed throughout the next sections in this chapter that $f_\mathbf{i} \neq \alpha$ at all data points. If this condition is violated for a particular datum, the corresponding function value is incremented (or decremented) by $\epsilon \ll \max\{f_\mathbf{i}\} - \min\{f_\mathbf{i}\}$. This is inevitable because the number of special cases which must otherwise be taken care of is simply tremendous.

The approach described in [Lorensen & Cline '87] assumes that the underlying

trivariate function f varies linearly along the edges of each cell: if $\mathbf{y}_0$ and $\mathbf{y}_1$ are the end points of an edge with corresponding function values f_0 and f_1, then

$$(\mathbf{x}^T, f)(t) = \big(\mathbf{x}^T(t), f(t)\big) = (1-t)({\mathbf{y}_0}^T, f_0) + t({\mathbf{y}_1}^T, f_1), \quad t \in [0,1].$$

If a contour intersects an edge ($f(t) = \alpha, t \in (0,1)$) the corresponding point $\mathbf{x}(t)$ is determined.

All points found on the edges of a cell are finally connected, thus forming closed polygons on the faces of a cell. These (non-planar) polygons are then triangulated. Using Lorensen's cell-by-cell method does not guarantee a continuous triangulation throughout the convex hull of the data set X (see [Dürst '88]). The reason for this is an inconsistency in constructing the polygons on a cell face shared by two neighbor cells: if four contour points are found on a face shared by two cells, they might be connected differently when the second of the two cells is considered. This is illustrated in Figure 3.2. (black dots at cell corners representing function values greater than the contour level α).

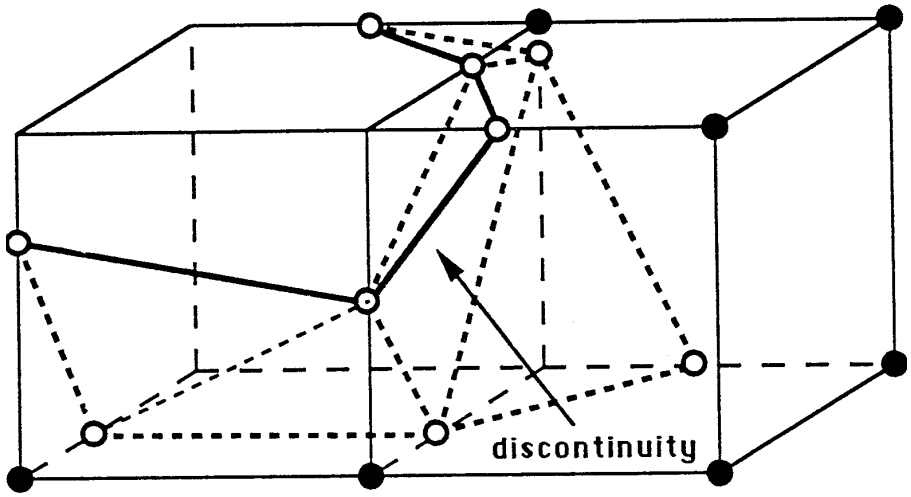

Fig. 3.2. Discontinuous piecewise planar contour approximation.

One solution to this problem is to subdivide a cell $C_{\mathbf{i}}$ into a set of tetrahedra whose union is $C_{\mathbf{i}}$ and whose intersections are tetrahedral faces. One way to split a cell $C_{\mathbf{i}}$ is to partition it into six tetrahedra:

$$
\begin{aligned}
T_{\mathbf{i}}^1 &= \{\ \mathbf{x} \mid \mathbf{x}(\mathbf{u}) = u_1\mathbf{x}_{i,j,k} + u_2\mathbf{x}_{i+1,j,k} + u_3\mathbf{x}_{i,j+1,k} + u_4\mathbf{x}_{i,j,k+1}\ \},\\
T_{\mathbf{i}}^2 &= \{\ \mathbf{x} \mid \mathbf{x}(\mathbf{u}) = u_1\mathbf{x}_{i+1,j,k} + u_2\mathbf{x}_{i,j+1,k} + u_3\mathbf{x}_{i,j,k+1} + u_4\mathbf{x}_{i,j+1,k+1}\ \},\\
T_{\mathbf{i}}^3 &= \{\ \mathbf{x} \mid \mathbf{x}(\mathbf{u}) = u_1\mathbf{x}_{i+1,j,k} + u_2\mathbf{x}_{i+1,j,k+1} + u_3\mathbf{x}_{i,j,k+1} + u_4\mathbf{x}_{i,j+1,k+1}\ \},\\
T_{\mathbf{i}}^4 &= \{\ \mathbf{x} \mid \mathbf{x}(\mathbf{u}) = u_1\mathbf{x}_{i+1,j,k} + u_2\mathbf{x}_{i+1,j+1,k} + u_3\mathbf{x}_{i,j+1,k} + u_4\mathbf{x}_{i,j+1,k+1}\ \},\\
T_{\mathbf{i}}^5 &= \{\ \mathbf{x} \mid \mathbf{x}(\mathbf{u}) = u_1\mathbf{x}_{i+1,j,k} + u_2\mathbf{x}_{i+1,j+1,k} + u_3\mathbf{x}_{i+1,j,k+1} + u_4\mathbf{x}_{i,j+1,k+1}\ \},\\
T_{\mathbf{i}}^6 &= \{\ \mathbf{x} \mid \mathbf{x}(\mathbf{u}) = u_1\mathbf{x}_{i+1,j+1,k} + u_2\mathbf{x}_{i+1,j,k+1} + u_3\mathbf{x}_{i,j+1,k+1} + u_4\mathbf{x}_{i+1,j+1,k+1}\ \},
\end{aligned}
$$

where $\sum_{l=1}^{4} u_l = 1$, $u_l \geq 0$ (barycentric coordinates). Assuming that f is a linear polynomial over each tetrahedron, $f(\mathbf{u}) = \sum_{l=1}^{4} u_l f_l$, $\mathbf{u}$ representing the barycentric coordinates of a point $\mathbf{x}$ in a tetrahedron with function values f_l at its four vertices, the contour of f is planar whenever it passes through the interior of a tetrahedron (see [Bloomquist '90] or [Foley & Lane '90]).

Both Lorensen's and the tetrahedral split-technique yield precise contours if the underlying function f originally is a linear polynomial defined over $\mathbb{R}^3$, $f(\mathbf{x}) = \sum_{|\mathbf{l}|\leq 1} c_{\mathbf{i}}\mathbf{x}^{\mathbf{l}}$, $|\mathbf{l}| = i + j + k$, $c_l \in \mathbb{R}$, $\mathbf{x}^{\mathbf{l}} = x^i y^j z^k$. However, if one prefers to avoid the tetrahedral split-approach and derive a piecewise planar contour approximation from the cells themselves, Lorensen's method must be modified in order to achieve a continuous triangulation.

The data for a cell are interpolated over the whole cell using an appropriate and simple interpolation method.

Definition 3.13. Let $C_{\mathbf{i}}$ be a cell; the associated **trilinear cell interpolant** is the trivariate polynomial

$$\begin{aligned}(\mathbf{x}^T, f)(u,v,w) &= \big(\mathbf{x}^T(u,v,w), f(u,v,w)\big) \\ &= \sum_{t=0}^{1}\sum_{s=0}^{1}\sum_{r=0}^{1} (\mathbf{x_r}^T, f_{\mathbf{r}})^{[\mathbf{i}]}\ B_r^1(u)B_s^1(v)B_t^1(w), \end{aligned} \tag{3.4.}$$

where $(\mathbf{x_r}^T, f_{\mathbf{r}})^{[\mathbf{i}]} = (\mathbf{x}_{i+r,j+s,k+t}{}^T, f_{i+r,j+s,k+t})$ are so-called Bézier points, $B_l^1(t) = (1-t)^{1-l}t^l$, $t \in [0,1]$, $l = 0,1$, are the Bernstein polynomials of degree one and $u,v,w \in [0,1]$.

Theorem 3.2. *The component $f(u,v,w)$ of the trilinear cell interpolant is a linear polynomial along each cell edge and a bilinear polynomial over each cell face.*

Proof. It is

$$f(u,v,w) = \textstyle\sum_{r=0}^{1} f_{\mathbf{r}}^{[\mathbf{i}]}\ B_r^1(u)$$

for $v,w \in \{0,1\}$, $u \in [0,1]$, $s,t \in \{0,1\}$ (analogous for the other edges, given by $u,w \in \{0,1\}$, $v \in [0,1]$, $r,t \in \{0,1\}$ and $u,v \in \{0,1\}$, $w \in [0,1]$, $r,s \in \{0,1\}$), and it is

$$f(u,v,w) = \textstyle\sum_{s=0}^{1}\sum_{r=0}^{1} f_{\mathbf{r}}^{[\mathbf{i}]} B_r^1(u)B_s^1(v)$$

for $w \in \{0,1\}$, $u,v \in [0,1]$, $t \in \{0,1\}$ (analogous for the other faces, $v \in \{0,1\}$, $u,w \in [0,1]$, $s \in \{0,1\}$, and $u \in \{0,1\}$, $v,w \in [0,1]$, $r \in \{0,1\}$).

q.e.d.

If a face shared by two cells contains two contour points on two edges of that face, only one possibility exists to connect them by a line segment. If there are four contour points (one on each edge of a face), an ambiguity arises for connecting

pairs of points on that face to construct two line segments. It is this case that leads to discontinuities in the contour triangulation obtained from Lorensen's technique. The trilinear cell interpolant solves this problem. Denoting the four corner data on a cell face by $(\mathbf{x}_{i,j}{}^T, f_{i,j})$, $i,j \in \{0,1\}$, one is concerned with the ambiguous case if

$$f_{0,0}, f_{1,1} > (<)\ \alpha \quad \text{and} \quad f_{1,0}, f_{0,1} < (>)\ \alpha. \tag{3.5.}$$

The contour points on the edges are again obtained from linear interpolation along the edges, consistent connections between them are assured by considering the contour

$$f(u,v) \ = \ \sum_{j=0}^{1}\sum_{i=0}^{1} \ f_{i,j} \ B_i^1(u)B_j^1(v) \ = \ \alpha, \tag{3.6.}$$

$u, v \in [0,1]$, over the whole face. Equation (3.6.) is equivalent to the equation

$$f(u,v) \ = \ \sum_{j=0}^{1}\sum_{i=0}^{1} \ \triangle^{i,j} f_{0,0} \ u^i v^j \ = \ \alpha, \tag{3.7.}$$

where

$$\triangle^{1,0} f_{0,0} = f_{1,0} - f_{0,0}, \quad \triangle^{0,1} f_{0,0} = f_{0,1} - f_{0,0} \quad \text{and}$$

$$\triangle^{1,1} f_{0,0} = \triangle^{1,0}(f_{0,1} - f_{0,0}) = (f_{1,1} - f_{1,0}) - (f_{0,1} - f_{0,0})$$

are the forward differences for two indices.

Theorem 3.3. *The contour defined by equation* (3.7.) *is a hyperbola with asymptotes given by*

$$u = u_0 = -\frac{\triangle^{0,1} f_{0,0}}{\triangle^{1,1} f_{0,0}} \quad \textit{and} \quad v = v_0 = -\frac{\triangle^{1,0} f_{0,0}}{\triangle^{1,1} f_{0,0}}, \quad u_0, v_0 \in [0,1]. \tag{3.8.}$$

Proof. Let $\triangle^{i,j}$ be the abbreviation for $\triangle^{i,j} f_{0,0}$ and equation (3.5.) be satisfied (ambiguous case). The asymptotic behavior for $f(u,v) = \alpha$ is proven quite easily:

$$\lim_{v\to\infty} u(v) = \lim_{v\to\infty} \frac{\alpha - \triangle^{0,0} - \triangle^{0,1} v}{\triangle^{1,0} + \triangle^{1,1} v} = -\frac{\triangle^{0,1}}{\triangle^{1,1}},$$

$$\lim_{u\to\infty} v(u) = \lim_{u\to\infty} \frac{\alpha - \triangle^{0,0} - \triangle^{1,0} u}{\triangle^{0,1} + \triangle^{1,1} u} = -\frac{\triangle^{1,0}}{\triangle^{1,1}}.$$

By performing an appropriate coordinate transformation it can be shown that $f(u,v) = \alpha$ is a hyperbola. A new coordinate system $\overline{S}$ is defined by its origin (u_0, v_0) and $\frac{\sqrt{2}}{2}(1,-1)$ and $\frac{\sqrt{2}}{2}(1,1)$ as its two orthogonal unit vectors determining a right-handed system. A point (u,v) is linearly mapped by the composition of a translation by $-(u_0,v_0)$ followed by a rotation by $-\frac{\pi}{4}$ onto the point $(\overline{u},\overline{v})$,

$$\overline{u} = \tfrac{\sqrt{2}}{2} \ \big(\ (u-u_0) + (v-v_0) \ \big),$$

$$\overline{v} = \tfrac{\sqrt{2}}{2} \ \big(\ -(u-u_0) + (v-v_0) \ \big).$$

The inverse map is given by

$$u = \tfrac{\sqrt{2}}{2} \ (\ \overline{u} - \overline{v} \ ) \ + \ u_0,$$

$$v = \tfrac{\sqrt{2}}{2} \ (\ \overline{u} + \overline{v} \ ) \ + \ v_0.$$

Expressing the function f in terms of $\overline{u}$ and $\overline{v}$ and inserting it into equation (3.7.) yields the equation of a hyperbola in standard position:

$$\overline{u}^2 - \overline{v}^2 \ = \ \tfrac{2}{(\triangle^{1,1})^2} \ \big(\ \triangle^{1,1}(\alpha - \triangle^{0,0}) + \triangle^{1,0}\triangle^{0,1} \ \big) \ = \ (-) \ a^2,$$

which is equivalent to

$$\frac{\overline{u}^2}{a^2} - \frac{\overline{v}^2}{a^2} \ = \ 1 \ \left(\frac{\overline{v}^2}{a^2} - \frac{\overline{u}^2}{a^2} \ = \ 1 \right), \qquad a \neq 0. \tag{3.9}$$

q.e.d.

It is now obvious how to derive a criterion for a proper connection of pairs of contour points on a cell face in the ambiguous case. The asymptotes $u = u_0$ and $v = v_0$ define four quadrants in the $uv-$domain square $[0,1]^2$, namely

$$[0,u_0] \times [0,v_0], \quad [u_0,1] \times [0,v_0],$$

$$[0,u_0] \times [v_0,1], \quad \text{and} \quad [u_0,1] \times [v_0,1].$$

Two contour points are connected to form a line segment if they lie in the same quadrant. The problem for the special case that the contour $f(u,v) = \alpha$ coincides with the two straight lines $u = u_0$ and $v = v_0$ (which is the case when equation (3.9.) collapses to $\overline{u}^2 = \overline{v}^2$) still remains. The ambiguity can be solved in two different ways. Either, one decides to connect pairs of contour points on opposite edges on the face (which is in accordance with the fact that the contour actually consists of two straight lines intersecting somewhere in the face's interior), or, one chooses an adhoc solution: connect pairs of contour points such that the quadrants in which the constructed line segments are lying in satisfy the condition to contain a corner (i,j), $i,j \in \{0,1\}$, for which $f(i,j) > \alpha$. Connecting pairs of points on opposite edges might lead to problems in the triangulation process of the constructed contour polygons later on, thus making the adhoc solution preferable.

In Figure 3.3., the ambiguous case is shown. The trilinear cell interpolant is restricted to a single cell face whose edges all yield a point on the contour $f(u,v) = \alpha$ (corner ordinates drawn as black dots representing function values greater than the contour level α). Contour points are drawn as circles, their connection is based on the asymptotes $u = u_0$ and $v = v_0$ of $f(u,v) = \alpha$.

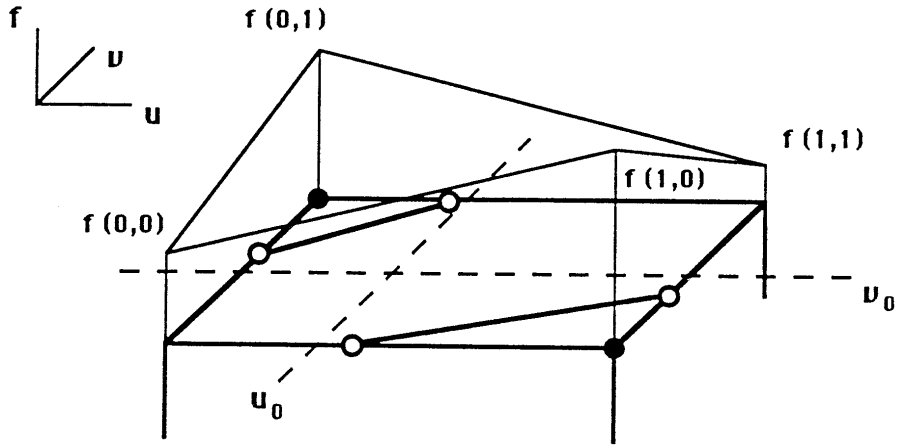

Fig. 3.3. Trilinear cell interpolant restricted to a cell face, solution for the ambiguous case.

Connecting all contour points found along the edges of a cell obviously yields a set of usually non-planar, closed polygons consisting of three (minimal number) to twelve (maximal number) of vertices. A polygon of length twelve can only be created if there are four contour points on each cell face, and all line segments belong to the same polygon. These polygons are interpreted as polygonal boundaries of a piecewise planar (triangular) approximation of a contour of a trivariate function with respect to a particular cell. Therefore, points of each polygon must be connected in order to constitute a contour triangulation inside a cell. Consistency constraints with respect to cells sharing faces require that the following condition is always satisfied:

Condition 3.1. *The only edges on cell faces in the contour triangulation are the line segments constituting the closed polygons constructed over the cells' faces. No other edges connecting contour points on cell faces are allowed.*

This rule guarantees that triangles completely lying on a cell face are never constructed. Only in the case that a cell face contains four contour points belonging to the same polygon one must assure that the above condition is not violated.

Theorem 3.4. (i) *If a closed contour polygon P consisting of the line segments $\overline{\mathbf{y}_i\mathbf{y}_{i+1}}$, $i = 0...(n-1)$, indices mod n, has at most one line segment on each cell face, every triangulation of P satisfies condition* 3.1.

(ii) *If a closed contour polygon has two line segments on the same face of a cell, condition* 3.1. *is violated by at least one triangulation of P.*

Proof. (i) All edges besides the line segments constituting P additionally needed for any of P's triangulations necessarily pass through the cells' interior.

(ii) If $\overline{\mathbf{y}_k\mathbf{y}_{k+1}}$ and $\overline{\mathbf{y}_l\mathbf{y}_{l+1}}$ are two line segments on the same face constituting P, there is at least one triangulation of P with the edge $\overline{\mathbf{y}_k\mathbf{y}_l}$

q.e.d.

Cells containing polygons whose triangulation might lead to a violation of condition 3.1. are illustrated in Figure 3.4. Polygons of length six, eight, nine, and twelve are shown. Black dots represent function values greater than α, circles are the polygons' vertices.

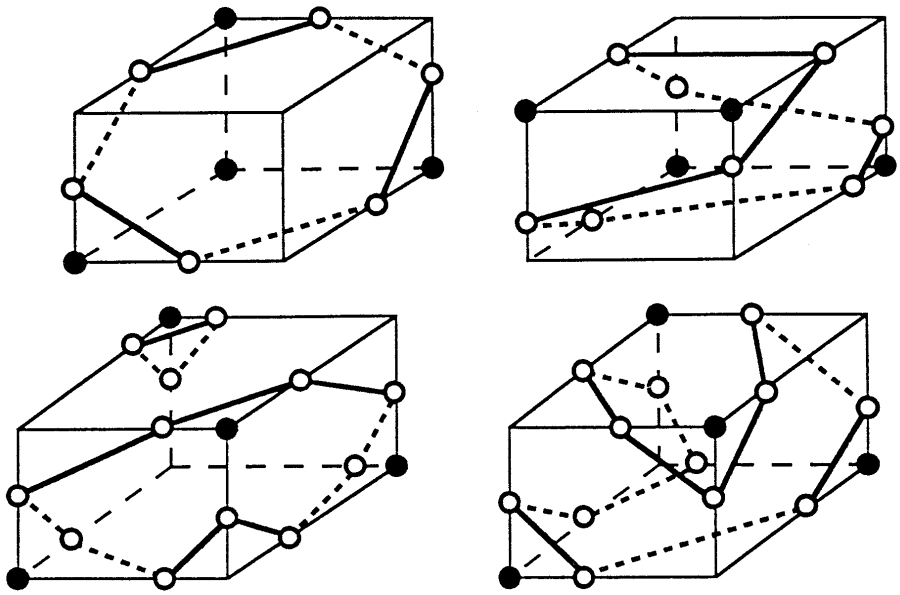

Fig. 3.4. Cells containing contour polygons
of length six, eight, nine, and twelve.

Theorem 3.5. *Let P be a closed contour polygon with line segments $\overline{\mathbf{y}_i\mathbf{y}_{i+1}}$, $i = 0...(n-1)$, with four contour points on at least one cell face F. Let $\mathbf{y}_n$ be a point not in the same plane as F. Then the point set $\{\mathbf{y}_0, ..., \mathbf{y}_n\}$ can be triangulated using only P's line segments without any other edges on F than two of P's line segments.*

Proof. The triangulation $\mathcal{T} = \left\{ (i, i+1, n) \mid i = 0...(n-1), \text{ indices mod } n \right\}$ is a triangulation not violating condition 3.1.

q.e.d.

An appropriate choice for $\mathbf{y}_n$ must be made in the case that a contour polygon has four points on the same cell face. The obvious way to choose $\mathbf{y}_n$ is to calculate a point on the contour $f(\mathbf{u}) = \alpha$.

Theorem 3.6. *Let P be a contour polygon in a cell $C_{\mathbf{i}}$ with four points on the cell face F_L (without loss of generality) given by*

$$F_L \;=\; [y_j, y_{j+1}] \times [z_k, z_{k+1}], \quad x = x_i.$$

Let P consist of the line segments $\overline{\mathbf{y}_i\mathbf{y}_{i+1}}$, $i = 0...(n-1)$, indices mod n, and $[0,1]^3$ be the associated domain in uvw-space and

$$\big\{\ (u = 0, v = v_0, w) \mid v_0 \in (0,1),\ w \in I\!R\ \big\} \qquad \textit{and}$$

$$\big\{\ (u = 0, v, w = w_0) \mid w_0 \in (0,1),\ v \in I\!R\ \big\}$$

be the two asymptotes for the hyperbola $f(\mathbf{u}) = \alpha$ (equation (3.8.) on F_L's corresponding face in uvw-space. Then, the point

$$\mathbf{y}_n\big(\mathbf{u}(t_0)\big) \;=\; \mathbf{y}_n\big((0, v_0, w_0) + t_0(1,0,0)\big), \tag{3.10.}$$

where

$$t_0 = \frac{\alpha - \sum_{k=0}^{1}\sum_{j=0}^{1} \triangle^{0,j,k} f_{0,0,0}\ v_0^j w_0^k}{\sum_{k=0}^{1}\sum_{j=0}^{1} \triangle^{1,j,k} f_{0,0,0}\ v_0^j w_0^k,} \tag{3.11.}$$

is a point on the contour $f(\mathbf{u}) = \alpha$.

Proof. Calculating the intersection of the line

$$\mathbf{u}(t) \;=\; (0, v_0, w_0) + t(1,0,0)$$

$t \in I\!R$, in $uvw-$space and the contour $f(\mathbf{u}) = \alpha$ of the trilinear cell interpolant

$$f(\mathbf{u}) \;=\; \sum_{k=0}^{1}\sum_{j=0}^{1}\sum_{i=0}^{1} \triangle^{i,j,k} f_{0,0,0}\ u^i v^j w^k \;=\; \alpha$$

and abbreviating the forward differences for three indices as $\triangle^{i,j,k} = \triangle^{i,j,k} f_{0,0,0}$ yields

$$t_0 = \frac{\alpha - (\triangle^{0,0,0} + \triangle^{0,1,0} v_0 + \triangle^{0,0,1} w_0 + \triangle^{0,1,1} v_0 w_0)}{\triangle^{1,0,0} + \triangle^{1,1,0} v_0 + \triangle^{1,0,1} w_0 + \triangle^{1,1,1} v_0 w_0}$$

$$= \frac{\alpha - \sum_{k=0}^{1} \sum_{j=0}^{1} \triangle^{0,j,k} f_{0,0,0} \; v_0^j w_0^k}{\sum_{k=0}^{1} \sum_{j=0}^{1} \triangle^{1,j,k} f_{0,0,0} \; v_0^j w_0^k}$$

determining a point in $xyz-$space serving as the additional point $\mathbf{y}_n$.

q.e.d.

Remark 3.9. The denominator in equation (3.11.) must not vanish. If it does vanish, the centroid of all contour points $\mathbf{y}_i$, $i = 0...(n-1)$,

$$\mathbf{y}_n = \frac{1}{n} \sum_{i=0}^{n-1} \mathbf{y}_i,$$

is chosen. This choice guarantees that $\mathbf{y}_n$ always is a point in the cell's interior.

Remark 3.10. The additional contour point $\mathbf{y}_n$ does not necessarily lie in a cell's interior (using the method from Theorem 3.6.). This might eventually lead to a contour triangulation violating (iii) in Definition 3.8. (the intersection of the interior of two triangles must be empty).

Figure 3.5. shows the exact and the piecewise linearly approximated contour $f(\mathbf{x}) = 1.5$ using the trilinear approach described above including the construction of an additional contour point $\mathbf{y}_n$ in a single domain cell's interior for the trilinear function

$$f\ (\mathbf{x}) \ = \ 2\ (1-x)(1-y)(1-z) + 1.6\ x(1-y)(1-z) + 1.4\ (1-x)y(1-z)$$

$$+1.4\ (1-x)(1-y)z + .4\ x(1-y)z + 2\ (1-x)yz + 2\ xyz,$$

where the cell is given by $[0,1] \times [0,1] \times [0,1]$.

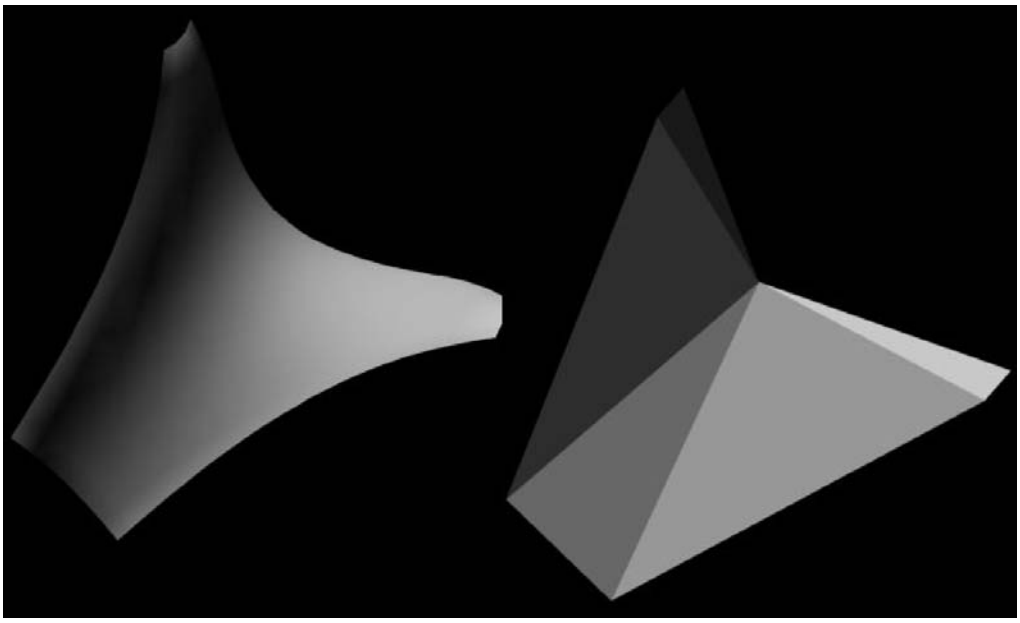

Fig. 3.5. Exact and piecewise linearly approximated contour in a cell,
$f(x,y,z) \;=\; 2(1-x)(1-y)(1-z) + 1.6x(1-y)(1-z) + 1.4(1-x)y(1-z)$
$+1.4(1-x)(1-y)z + .4x(1-y)z + 2(1-x)yz + 2xyz \;=\; 1.5,\; x,y,z \in [0,1].$

Remark 3.11. The piecewise planar contour approximation is trilinearly precise with respect to a single cell in the sense that all the contour points used for the approximation are points of a contour of the trilinear cell interpolant. By construction it is a continuous two-dimensional contour triangulation in the sense of the Definitions (3.8.) and (3.11.).

Remark 3.12. The problem of consistently connecting contour points on cells' faces does not arise when using convex polyhedra having triangular faces only. Therefore, it might be worth considering a decomposition of a subset of $\mathbb{R}^3$ into a set of octahedra as well. In this case, contour polygons would have maximally one line segment on a face of an octahedron (using a linear interpolation approach).

Figure 3.6. shows a piecewise triangular contour approximation for the function $f(\mathbf{x}) = 1.2\left((x-10)^2-(y-10)^2+(z-10)^2\right)$ and the contour level $f(\mathbf{x}) = 60$. The trivariate function is evaluated on an equidistantly spaced rectilinear data grid of $21 \cdot 21 \cdot 21$ points in $[0,20]^3$. All contour triangles are rendered using flat shading.

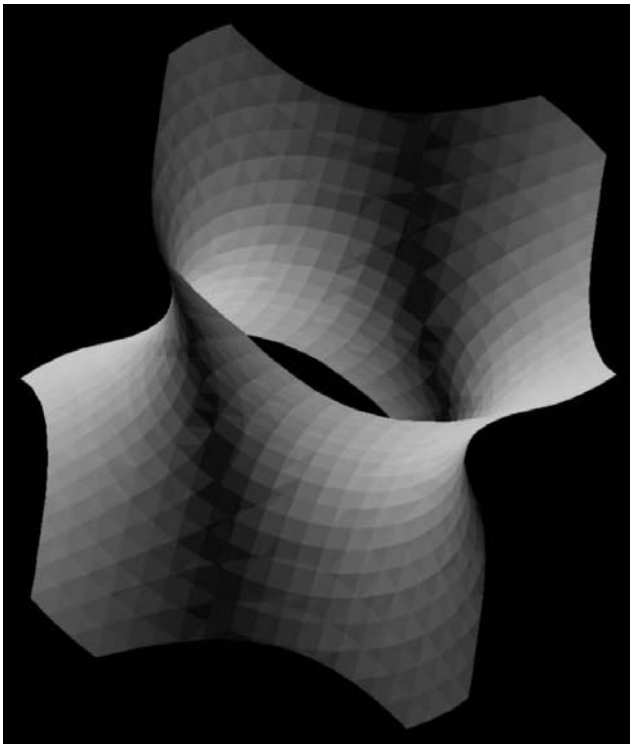

Fig. 3.6. Triangular approximation of contour level $f(x,y,z) = 60$ for $f(x,y,z) = 1.2\left((x-10)^2-(y-10)^2+(z-10)^2\right),\ \ x,y,z \in [0,20]$.

Remark 3.13. At this stage of the triangulation process the quality of the triangulation within a single cell is not taken into account. As soon as one has obtained the whole set of triangles approximating the contour throughout all cells, *smoothness* criteria can be used to improve the triangulation.

3.3. Computing topological information for a piecewise triangular trivariate contour approximation

From a computational point of view, a rectilinear trivariate data set is investigated iteratively, cell by cell, to generate a contour approximation. Considering a single cell three tables are established, the first table storing all the contour points with their three-dimensional coordinates (ordered),

$$Y \;=\; \{\; (i, \mathbf{x}_i{}^T) = (i, x_i, y_i, z_i) \mid i = 0...n_C - 1 \;\},$$

a second table defining which vertices in the first table constitute line segments of closed polygons on the cell's faces (ordered),

$$E \;=\; \{\; (j, v_j^1, v_j^2) \mid j = 0...n_C - 1 \;\},$$

v_j^1 and v_j^2 being indices referring to Y, and a third table (derived from E) specifying which line segments (ordered) form the edges of closed polygons on the cell's faces,

$$P \;=\; \{\; (p, e_p^0, e_p^1, ..., e_p^{n_p - 1}) \mid p = 0...m - 1 \;\},$$

e_p^k, $k = 0...n_p - 1$, being indices referring to E.

The three tables Y, E, and P are for temporary use only. As soon as a triangulation for all the closed contour polygons in the set P has been computed, the contour points from the temporary table Y are copied into a permanent, global vertex table V (ordered),

$$V \;=\; \{\; (i, \mathbf{x}_i{}^T) = (i, x_i, y_i, z_i) \mid i = 0...n_v - 1 \;\}, \qquad (3.12.)$$

containing all contour points found throughout the whole data set. The triangles constructed in a single cell's interior are also added to a permanent, global table $\mathcal{T}$

defining the overall triangular contour approximation,

$$\mathcal{T} = \{\ T_t = (t, v_t^1, v_t^2, v_t^3) \mid t = 0...n_t - 1\ \}. \tag{3.13.}$$

Having computed the complete set $\mathcal{T}$ of all triangles approximating a certain contour, it is essential to derive topological information still "hidden" in the triangulation. Data reduction algorithms and surface generation schemes commonly require neighborhood information. An algorithm is given that computes the neighbors of each triangle considering the table $\mathcal{T}$ only. The contour approximation for $f(\mathbf{u}) = \alpha$ might be split into several non-connected parts, as mentioned before. Therefore, it is also necessary to know to which part a particular triangle belongs to if one wants to model the different contour parts separately.

Algorithm 3.1. determines the neighbors for each triangle, i.e., for a given triangle T_t the algorithm generates the indices of its maximal three neighbors in $\mathcal{T}$.

Algorithm 3.1. *Neighborhood*

Input: table $\mathcal{T}$ of triangles T_t, each given by its own index (referring to $\mathcal{T}$) and its three vertex indices (referring to V).
Output: number of neighbor triangles and their indices (referring to $\mathcal{T}$) for each triangle T_t in $\mathcal{T}$.

```
for i = 0 to n_t − 1
    (   cnt := 0;    /* number of neighbors */
        j := 0;
        while j < n_t and cnt < 3
            (   if  i ≠ j and T_i and T_j are neighbors
                    then
                    (   cnt := cnt + 1;
                        cnt − th neighbor of T_i := j;
                    )
                j := j + 1;
            )
        number of neighbors of T_i := cnt;
    )
```

This algorithm is of order $O({n_t}^2)$ with respect to the total number of triangles. Its performance can be improved by storing the index-triple (i, j, k) of the left-front-lower corner of the cell $C_{\mathbf{i}}$ for each triangle T_t when T_t lies in $C_{\mathbf{i}}$'s interior. Thus, the search for the neighbors of a particular triangle inside a cell $C_{\mathbf{i}}$ can be restricted to the cell $C_{\mathbf{i}}$ itself and its six neighbor cells, $C_{i-1,j,k}$, $C_{i+1,j,k}$, $C_{i,j-1,k}$, $C_{i,j+1,k}$, $C_{i,j,k-1}$, and $C_{i,j,k+1}$. Hence, order $O(n_t)$ can be achieved.

Two triangles in $\mathcal{T}$ belong to the same part of the contour approximation if there is a path from one triangle to the other one such that all triangles constituting this path determine pairs of neighbor triangles (Definition 3.9.). To effectively compute the part (index) a particular triangle belongs to, the following algorithm is used:

At the beginning all triangles share the not-yet-assigned part index -1. The first triangle T_0 in $\mathcal{T}$ is assigned to the first part with index 0. For all the other triangles T_t, one checks among its neighbors, whether at least one of them has already been assigned to some part. If none of the neighbors has been assigned yet, a new part (index) is introduced for triangle T_t; if at least one among its neighbors already belongs to a certain part, the minimal part (index) among all T_t's assigned neighbors is selected as T_t's part (index). If the neighbors of T_t which are assigned to a part do not all agree with this minimal part (index), all triangles in $\mathcal{T}$ assigned to such a different part (index) must now also be assigned to the selected minimal part (index).

Algorithm 3.2. *Part of contour*

```
Input:   table 𝒯 of triangles T_t (including the neighborhood information).
Output:  part index for each triangle which determines the part
         of the contour approximation it belongs to.

/* initially all triangles have a not-yet-assigned part index −1 */
p := 0;     /* first valid part index */
for t = 0 to n_t − 1
    (   determine minimal part index min among
        all the part indices of T_t's neighbors;
        if min = −1
            then
            (   part index for triangle T_t := p;
                p := p + 1;    /* another part of contour introduced */
            )
            else
            (   part index for triangle T_t := min;
                if there is 1 [are 2] valid part index p̄ [indices p̄, p̿] ≠ min
                  among T_t's neighbors
                   then
                   (   part index for all triangles assigned to p̄ [p̄, p̿] := min;
                                          /* connecting triangle has been found */
                       p := p − 1 [2];    /* reduce number of parts appropriately */
                   )
            )
    )
```

The case in brackets ("[ ]") in algorithm 3.2. indicates the situation when a triangle has three neighbors with valid part indices ($\neq -1$) which are all different from each other. At the end, each triangle is assigned to a certain part of the triangular contour approximation. Obviously, algorithm 3.2. is of order $O(n_t)$.

The principal of selecting a part (index) for a triangle is shown in Figure 3.7. The minimal part (index) among T_5's neighbors is 1. Therefore, T_5 as well as all triangles belonging to part 2 are assigned to part 1.

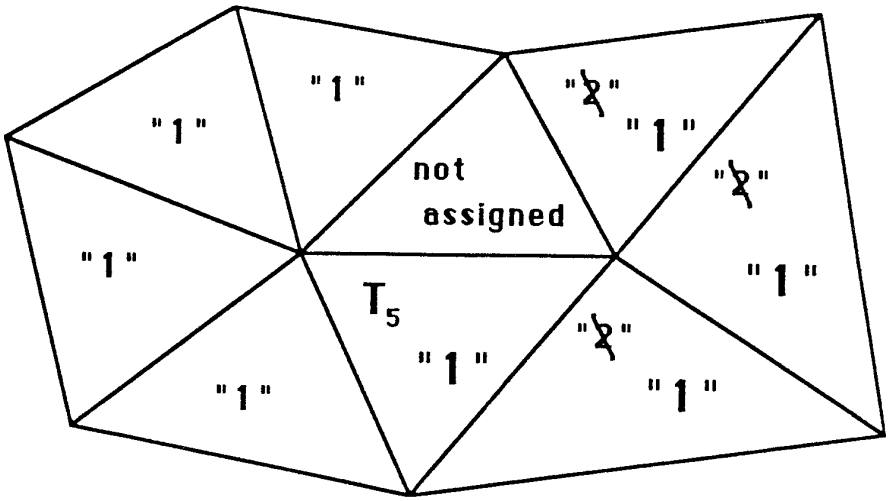

Fig. 3.7. Assigning the part index to a triangle in a contour triangulation.

Remark 3.14. It might be worth considering some methods for improving the triangulation of the contour approximation at this point. Knowing the neighbors for each triangle in $\mathcal{T}$, the "$max - min$" or "$min - max$" angle criteria could be used to iteratively eliminate triangles with small angles ([Cline & Renka '84], [Lawson '77]). Common algorithms swap diagonals of quadrilaterals (given by two neighbor triangles) in order to enhance the angle configuration.

Considering the fact that the triangulation to be improved is not a planar triangulation, different optimization criteria might be appropriate. An algorithm to increase the smoothness of a two-dimensional triangulation in three-dimensional space is described in [Choi et al. '88]. There, the (local) quality measure of a trian-

gulation is the angle between normal vectors of neighbor triangles. The objective is to minimize these angles by swapping diagonals of quadrilaterals.

3.4. Gradient approximation for rectilinear data

It is not the purpose of this chapter to discuss or derive new ways for gradient/normal approximation in full detail. Rather, the principal problem is stated, and general solutions are reviewed in a more survey fashion. Gradient/normal information is necessary for curvature approximation, data reduction, and surface generation, discussed in the following chapters. The quality of these subsequent modeling steps is very much dependent on the quality of the gradient/normal estimation.

In order to construct trivariate functions approximating trivariate data sets given in either rectilinear or scattered form or to generate smooth surfaces fitting the different parts of a given two-dimensional contour triangulation of some trivariate function, gradients and normals must be estimated if positional information is available only. In the case of normal vector approximation, outward unit normal vectors are estimated defining oriented tangent planes for all contour points in the contour triangulation.

General information about the construction of bivariate/trivariate functions approximating scattered data can be found in [Alfeld '89], [Barnhill '85], [Foley '87], [Franke & Nielson '91], [Hoschek & Lasser '89], [Nielson & Franke '83], and [Worsey & Farin '87]. In [Stead '84] different schemes are compared for estimating

gradients. A local operator generating normal vector estimates for rectilinear a data set in an "optimal" sense is described in [Zucker & Hummel '81].

With respect to the strategy pursued here, it is of greater interest to approximate normal vectors since it is a two-dimensional contour triangulation which will be modelled later on. One can choose among two possibilities how to obtain normal vector estimates. The first possibility consists of constructing a trivariate function locally approximating the rectilinear/scattered data, hence also defining gradients at the points in a contour triangulation. The second possibility rather derives normal vector estimates from a contour triangulation directly.

If the first possibility is chosen, it is important to realize that the gradients at contour points also determine normal vectors:

Theorem 3.7. *Let* $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ *be a* C^1 *function and* $\bigtriangledown f(\mathbf{x}_0)$ *be non-vanishing. Let* $\mathbf{x}_0$ *be a point in* $C_f(\alpha)$ *and* $\mathbf{v}$ *any tangent vector to* $C_f(\alpha)$ *at* $\mathbf{x}_0$*; then* $\bigtriangledown f(\mathbf{x}_0)$ *is normal to the contour* $C_f(\alpha)$ *at* $\mathbf{x}_0$*,*

$$\bigtriangledown f(\mathbf{x}_0)\ \mathbf{v}\ =\ 0. \tag{3.14.}$$

Proof. Let $\mathbf{c}(t) \subset C_f(\alpha)$ be a curve on the contour such that $\mathbf{c}(0) = \mathbf{x}_0$ and $\dot{\mathbf{c}}(0) = \mathbf{v}$; considering the fact that $f(\mathbf{c}(t)) = \alpha$, and using the chain rule yields

$$0 = \tfrac{d}{dt} f(\mathbf{c}(t))\big|_{t=0} = \bigtriangledown f(\mathbf{x}_0)\ \dot{\mathbf{c}}(0) = \bigtriangledown f(\mathbf{x}_0)\ \mathbf{v}.$$

q.e.d.

Definition 3.14. Let $f : I\!R^3 \rightarrow I\!R$ be a C^1 function and $\bigtriangledown f(\mathbf{x}_0)$ be non-vanishing. The **outward unit normal vector** to $C_f(\alpha)$ at $\mathbf{x}_0$ is the vector

$$\mathbf{n}_0 = \left(\bigtriangledown f(\mathbf{x}_0) \; / \; || \bigtriangledown f(\mathbf{x}_0)|| \right)^T, \tag{3.15.}$$

where $||(x, y, z)|| = \sqrt{(x^2 + y^2 + z^2)}$. The **oriented tangent plane** at $\mathbf{x}_0$ is given by the set of all points $\mathbf{x} = (x, y, z)^T \in I\!R^3$ satisfying the equation

$$\bigtriangledown f(\mathbf{x}_0) \; (x - x_0, y - y_0, z - z_0)^T \; = \; 0. \tag{3.16.}$$

Choosing the alternative of constructing an approximating function in a neighborhood around a contour point, one must be aware to keep the original rectilinear/scattered trivariate data set.

A method for normal estimation which has proven to yield rather good results is discussed in [Zucker & Hummel '81]. Summing up the approach, Zucker reduces normal estimation to a minimization problem. The expression minimized is

$$|| \; f(\mathbf{x}) - E_{\{a,b,c\}}(\mathbf{x}) \; ||.$$

Here, $f(\mathbf{x})$ is a known trivariate function defined over the unit ball B ($B = \left\{\mathbf{x} | x^2 + y^2 + z^2 \leq 1\right\}$) and $E_{\{a,b,c\}}(\mathbf{x})$ is the function

$$E_{\{a,b,c\}}(\mathbf{x}) \; = \; \begin{cases} +1, & \text{if } ax + by + cz \geq 0; \\ -1, & \text{otherwise.} \end{cases}$$

The coefficients a, b, and c are the unknowns defining an oriented plane through the origin with normal vector $\mathbf{n} = (a, b, c)^T$ used as the normal estimate. The norm is the L_2-norm,

$$||f(\mathbf{x})|| \;=\; \sqrt{\int\int\int_B \; f^2(\mathbf{x})\; dxdydz}.$$

The result is then applied to the discrete case, given a rectilinear data set with equal, equidistant spacing in all three spatial directions, $\Delta = \triangle x_i = \triangle y_j = \triangle z_k$, all i, j, k. A simple, local operator is derived in order to optimally approximate the outward normal vector at a grid point $\mathbf{x_i}$ fixed as the origin of a local coordinate system for the discrete minimization problem.

Theorem 3.8. *Considering solely the* 27 *neighbor data* $(\mathbf{x}_{i+r,j+s,k+t}, f_{i+r,j+s,k+t})$, $r, s, t \in \{-1, 0, 1\}$, *nearest to* $(\mathbf{x_i}^T, f_\mathbf{i})$ *in an equally, equidistantly spaced rectilinear data set, a normal vector* $\mathbf{n_i} = (nx_\mathbf{i}, ny_\mathbf{i}, nz_\mathbf{i})^T$ *at* $\mathbf{x_i}$ *is optimally approximated by*

$$\begin{aligned}
nx_\mathbf{i} &= \sum_{r\in\{-1,1\},\ s,t\in\{-1,0,1\}} r\ c_{i+r,j+s,k+t}\ f_{i+r,j+s,k+t},\\
ny_\mathbf{i} &= \sum_{s\in\{-1,1\},\ r,t\in\{-1,0,1\}} s\ c_{i+r,j+s,k+t}\ f_{i+r,j+s,k+t},\\
nz_\mathbf{i} &= \sum_{t\in\{-1,1\},\ r,s\in\{-1,0,1\}} t\ c_{i+r,j+s,k+t}\ f_{i+r,j+s,k+t}, \qquad (3.17.)
\end{aligned}$$

where $c_{i+r,j+s,k+t} = \frac{\sqrt{|r|+|s|+|t|}}{|r|+|s|+|t|}$, *subject to minimizing* $||\ f(\mathbf{x}) - E_{\{a,b,c\}}(\mathbf{x})\ ||$ *in this particular discrete case.*

Proof. See [Zucker & Hummel '81], pages 326 and 329-331.

Normalizing $\mathbf{n_i}$ yields the desired outward unit normal vector at the point $\mathbf{x_i}$. The principle for computing the x-coordinate of $\mathbf{n_i}$ is sketched in Figure 3.8. The involved function values and their weights for the normal vector approximation at a grid point $\mathbf{x_i}$ are shown using Zucker's "$3 \cdot 3 \cdot 3$" operator.

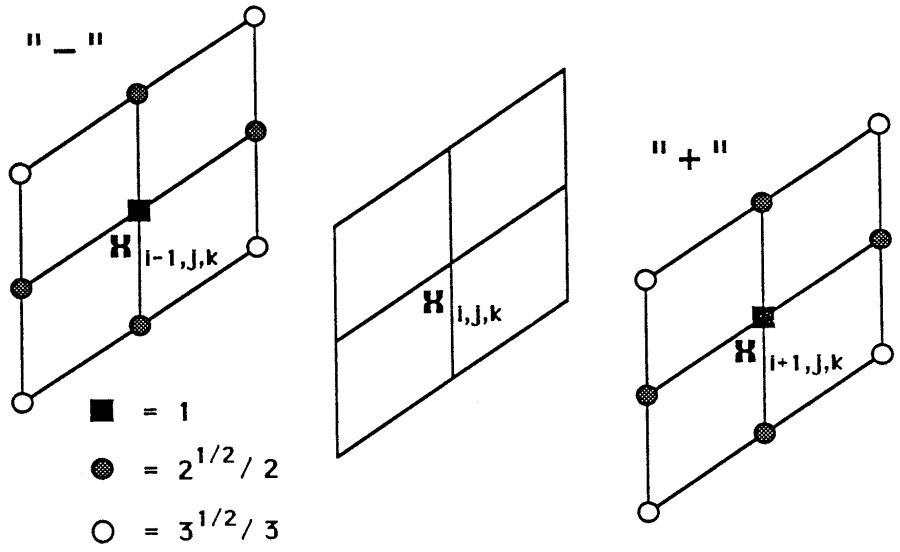

Fig. 3.8. The 18 function values and their weights needed for approximating the $x-$coordinate for a normal using Zucker's operator.

The normal vector for a contour point $\mathbf{x}_i \in V$ (formula (3.12.)) on an edge of a cell $C_\mathbf{i}$ is approximated by linear interpolation of the estimated normal vectors at the rectilinear grid points defining that edge, a normal vector in $C_\mathbf{i}$'s interior is approximated by trilinear interpolation of the eight normal vectors estimated for the cell's corner points. For the case that a contour point $\mathbf{x}_i$ is along the cell edge $\overline{\mathbf{ab}}$ one chooses the outward normal vector $\mathbf{n}_i$ at this point to be

$$\mathbf{n}_i = \mathbf{n}_i\big|_{\mathbf{x}=\mathbf{x}_i} = (1-t)\mathbf{n}\big|_{\mathbf{x}=\mathbf{a}} + t\mathbf{n}\big|_{\mathbf{x}=\mathbf{b}}, \quad t = \frac{||\mathbf{x}_i - \mathbf{a}||}{||\mathbf{b} - \mathbf{a}||}, \tag{3.18.}$$

where $t \in [0,1]$ and $||(x,y,z)^T||$ is the Euclidean norm. In the case that $\mathbf{x}_i$ is in the

interior of the cell $C_{\mathbf{i}}$ one chooses

$$\mathbf{n}_i = \mathbf{n}_i\big|_{\mathbf{x}=\mathbf{x}_i} = \sum_{t=0}^{1}\sum_{s=0}^{1}\sum_{r=0}^{1} \mathbf{n_r}\ B_r^1(u)B_s^1(v)B_t^1(w),$$

$$u = \frac{|x_i - \mathbf{x_i^x}|}{\Delta}, \quad v = \frac{|y_i - \mathbf{x_i^y}|}{\Delta}, \quad w = \frac{|z_i - \mathbf{x_i^z}|}{\Delta}, \tag{3.19.}$$

where $\mathbf{n_r} = \mathbf{n}_{i+r,j+s,k+t}$, $r, s, t \in \{0, 1\}$, are the eight outward normal vector estimates at $C_{\mathbf{i}}$'s corner points, where an equally, equidistantly spaced rectilinear point set is assumed, $\Delta = \triangle x_i = \triangle y_j = \triangle z_k$, all i, j, k, $\mathbf{x}_i = (x_i, y_i, z_i)^T$ is the contour point, $\mathbf{x_i} = (\mathbf{x_i^x}, \mathbf{x_i^y}, \mathbf{x_i^z})^T$ is the left-front-lower corner point of the cell $C_{\mathbf{i}}$, $B_l^1(t) = (1-t)^{1-l}t^l$, $t \in [0, 1]$, $l = 0, 1$, are the Bernstein polynomials of degree one and $u, v, w \in [0, 1]$.

Normalizing the estimates $\mathbf{n}_i$ finally determines the set of (ordered) outward unit normal vectors at each contour point,

$$N \ = \ \left\{ \ (i, {\mathbf{n}_i}^T) = (i, nx_i, ny_i, nz_i) \ | \ ||\mathbf{n}_i|| = 1, \ i = 0...n_v - 1 \ \right\}. \tag{3.20.}$$

Figure 3.9. is obtained from the same data set as the one used for Figure 2.1. CAT scan density measurements are given as an equally, equidistantly spaced rectilinear data set of $68 \cdot 64 \cdot 64$ points with associated density values $f_{\mathbf{i}} \in [0, 255]$. The contour level approximated is $f(\mathbf{x}) = 12.5$. The triangular approximation consists of almost $30,000$ contour points and $60,000$ triangles. Outward unit normal vectors for each contour point are estimated using Zucker's approach and required for this Gouraud-shaded rendering of the contour approximation.

Fig. 3.9. Human skull obtained from a rectilinear CAT scan data set, $68 \cdot 64 \cdot 64$ points, $f_{i,j,k} \in [0, 255]$, approximation for $f(x, y, z) = 12.5$.

Remark 3.15. The problem of estimating normal vectors for points in a two-dimensional triangulation in three-dimensional space using the triangulation alone has hardly been investigated. A possible good solution to the problem might be the following approach.

It is well known in differential geometry that a surface in three-dimensional space can locally be approximated by the graph of a differentiable bivariate function. In the case of a two-dimensional triangulation in three-dimensional space, one usually considers the points $\mathbf{y}_j \in V$, $j = 1...m$, (equation (3.12.)) determining an edge with a particular point $\mathbf{x}_i = \mathbf{y}_0 \in V$ in the triangulation as a localization of

the triangulation,

$$Y \;=\; \{\; \mathbf{x}_i = \mathbf{y}_0 \;\} \;\cup\; \{\; \mathbf{y}_j \mid \overline{\mathbf{x}_i\mathbf{y}_j} \text{ is an edge in the triangulation, } j = 1...m \;\}.$$

By introducing a local, right-handed coordinate system S, defined by $\mathbf{x}_i$ as origin and three mutually perpendicular unit vectors $\mathbf{d}_1$, $\mathbf{d}_2$, and $\mathbf{n}$, a plane P is defined by the origin and the first two unit vectors. The points in Y can now be projected into the plane P and their distances d_j, $j = 0...m$, from P be calculated. Assuming that all projected points in P are different, a bivariate function, e.g., a second degree polynomial, can be constructed using the least squares method to approximate the $m+1$ function values $f_j = d_j$, $j = 0...m$, considering the constraints

$$f(\overline{x}_j, \overline{y}_j) \;=\; \sum_{r+s+t\leq 2,\; r,s,t\geq 0} c_{r,s,t}\; \overline{x}_j{}^r \overline{y}_j{}^s \overline{z}_j{}^t \;=\; d_j, \qquad j = 0...m,$$

where $\overline{x}$ and $\overline{y}$ are the coordinates of a projected point in P with respect to the two-dimensional coordinates system defined by $\mathbf{x}_i$ (origin) and the unit vectors $\mathbf{d}_1$ and $\mathbf{d}_2$, and d_j is interpreted as the function value of f at the corresponding projected point in the direction of $\mathbf{n}$.

Assuming that the linear system of equations for the least squares solution does not imply a vanishing determinant, the unknown coefficients $c_{r,s,t}$ determine a residual vector $\mathbf{r}$ which can be measured using the L_2-norm,

$$\| \mathbf{r} \| \;=\; \sqrt{\sum_{j=0}^{m} \left(\; f(\overline{x}_j, \overline{y}_j) - d_j \;\right)^2}.$$

This expression really depends on the choice of the orientation of the system S. Changing the orientation of $\mathbf{n}$ (determining the f-axis) appropriately, might lead

to a minimization of $||\ \mathbf{r}\ ||$. Choosing the normal of the graph of f at $\mathbf{x}_i$ based on such an "optimally oriented" coordinate system S presumably is a good estimation for a normal vector $\mathbf{n}_i$. The direction for the normal vector is still ambiguous ($\mathbf{n}_i$ or $-\mathbf{n}_i$).

Remark 3.16. It is also worth considering a contour approximation for a finite data set G in either scattered or rectilinear form obtained by computing the length of the gradient estimates for each point in an original trivariate data set,

$$G \ = \ \left\{ \ ({\mathbf{x}_i}^T, g_i) = ({\mathbf{x}_i}^T, ||\bigtriangledown f(\mathbf{x}_i)||) \mid \mathbf{x}_i \in I\!R^3,\ g_i \in I\!R,\ i = 0...n \ \right\}.$$

This is a common approach in computer vision for edge detection. Boundaries of objects in an image (brightness or density functions) are characterized by large gradients.

Chapter 4

Curvature approximation for triangulated surfaces and trivariate functions

4.1. Introduction and essential terms of differential geometry

Methods for exactly calculating and approximating curvatures are important in geometric modeling for two reasons. In order to judge the quality of a surface one commonly computes curvatures for points on the surface, renders the surface's curvature as a texture map onto the surface and can thereby detect regions with undesired curvature behavior, such as surface regions locally changing from an elliptic to a hyperbolic shape. On the other hand, surface schemes are being developed requiring higher order geometric information as input, e.g., normal vectors and normal curvatures.

Definitions and theorems from classical differential geometry are reviewed as far as they are needed for the proceeding. In classical differential geometry a surface is understood as a mapping from $\mathbb{R}^2$ to $\mathbb{R}^3$,

$$\mathbf{x}\,(\mathbf{u}) \;=\; \big(\; x(u,v), y(u,v), z(u,v)\;\big)^T \;\subset\; \mathbb{R}^3, \qquad \mathbf{u} \in D \;\subset\; \mathbb{R}^2. \tag{4.1.}$$

The standard formulae are then used to derive techniques for approximating normal curvatures when a two-dimensional triangulation of a finite point set with associated outward unit normal vectors is given in three-dimensional space. Consequently, curvature estimates can be incorporated into existing surface generating schemes allowing curvature input. The quality of the curvature approximation is tested for

triangulated surfaces obtained from a known parametric surface $\mathbf{x}(\mathbf{u})$.

The theory of two-dimensional surfaces can easily be extended to the case of three-dimensional surfaces, e.g., graphs of trivariate functions approximating scalar fields over a three-dimensional domain,

$$\left(\mathbf{x}^T, f(\mathbf{x}) \right)^T = \left(x, y, z, f(x,y,z) \right)^T \subset I\!R^4, \quad \mathbf{x} \in D \subset I\!R^3. \tag{4.2.}$$

If the approximating function $f(\mathbf{x})$ is known, normal curvatures for its graph can be computed accurately, thus allowing to visualize the graph's curvature behavior using one of the rendering techniques for trivariate data sets introduced in chapter 2. Qualitative changes in f's three-dimensional graph in four-dimensional space can be observed, hence providing a quality measure for the chosen approximation method.

Future trivariate scattered data approximation schemes might as well require input such as normal curvatures when the approximation process is seen from a more geometric point of view interpreting the result as a three-dimensional hypersurface. An estimation method is presented for approximating normal curvatures at four-dimensional points $\left(x_i, y_i, z_i, f(x_i, y_i, z_i) \right)^T$ on a three-dimensional hypersurface in order to generate a smooth graph obtained by solving the trivariate approximation problem. Again, the quality of the curvature estimation technique is tested for known trivariate functions. Possibly, multivariate approximation schemes for even higher dimensions ($f(x_1, ..., x_n)$, $n \geq 4$) will consider such geometric information shortly.

Good introductions to differential geometry are [Brauner '81], [do Carmo '76], [Lipschutz '80], [Strubecker '55,'58,'59], and [Struik '61]. Differential geometry is treated more analytically in [O'Neill '69]. One of the most comprehensive works on this subject is [Spivak '70]. Some information can also be found in [Farin '88]. An example for estimating curvatures from a discrete point set is [Calladine '86]. There, a technique for approximating Gaussian curvature for points in a two-dimensional triangulation in three-dimensional space is discussed. An example for a surface scheme allowing curvature input is introduced in [Hagen & Pottmann '89]; a triangular surface scheme is described considering positional, normal vector, and normal curvature information.

Definition 4.1. A **regular parametric two-dimensional surface** of class C^m ($m \geq 1$) is the point set S in real three-dimensional space $\mathbb{R}^3$ defined by the mapping

$$\mathbf{x} \;=\; \mathbf{x}\,(\mathbf{u}) \;=\; \big(\; x(u,v), y(u,v), z(u,v) \;\big)^T \tag{4.3.}$$

of an open set $U \subset \mathbb{R}^2$ into $\mathbb{R}^3$ such that

(i) all partial derivatives of x, y, and z of order m or less
are continuous in U, and

(ii) $\mathbf{x}_u \times \mathbf{x}_v \neq (0,0,0)^T$ for all $(u,v) \in U$

(the subscripts u and v indicating partial differentiation with respect to u and v, respectively).

Since condition (ii) in Definition 4.1. implies the linear independence of the two

vectors $\mathbf{x}_u$ and $\mathbf{x}_v$ at any point on the regular surface, they determine the tangent plane at every surface point.

Definition 4.2. The **tangent plane** at a point $\mathbf{x}_0 = \mathbf{x}(\mathbf{u}_0)$ on a regular parametric two-dimensional surface in three-dimensional space is defined as the set of all points $\mathbf{y}$ in $I\!R^3$ satisfying the equation

$$\mathbf{y} \;=\; \mathbf{x}_0 + a\mathbf{x}_u(\mathbf{u}_0) + b\mathbf{x}_v(\mathbf{u}_0), \qquad a, b \in I\!R. \tag{4.4.}$$

Definition 4.3. The **outward unit normal vector** $\mathbf{n}_0 = \mathbf{n}(\mathbf{u}_0)$ of a regular parametric surface at a point $\mathbf{x}_0$ is given by

$$\mathbf{n}_0 \;=\; \frac{\mathbf{x}_u(\mathbf{u}_0) \times \mathbf{x}_v(\mathbf{u}_0)}{||\mathbf{x}_u(\mathbf{u}_0) \times \mathbf{x}_v(\mathbf{u}_0)||} \;=\; \frac{\mathbf{x}_u \times \mathbf{x}_v}{||\mathbf{x}_u \times \mathbf{x}_v||}, \tag{4.5.}$$

where $||\cdot||$ indicates the Euclidean norm.

Definition 4.4. Let $\mathbf{x}(\mathbf{u})$ be a regular parametric surface of class m, $m \geq 2$, and $\mathbf{c}(t) = \mathbf{c}\big(u(t), v(t)\big)$ be a (regular) curve of class 2 on the surface through the point $\mathbf{x}_0 = \mathbf{x}(\mathbf{u}_0)$. The **normal curvature vector** to $\mathbf{c}(t)$ at $\mathbf{x}_0$ is the projection of the curvature vector $\mathbf{k} = \dot{\mathbf{t}}/||\dot{\mathbf{t}}||$, $\mathbf{t} = \dot{\mathbf{c}}/||\dot{\mathbf{c}}||$, onto the unit surface normal vector $\mathbf{n}_0$,

$$\mathbf{k}_n \;=\; (\mathbf{k} \cdot \mathbf{n}_0)\; \mathbf{n}_0. \tag{4.6.}$$

The proportionality factor $\mathbf{k} \cdot \mathbf{n}_0$ is called the **normal curvature**, denoted by κ_n.

Definition 4.5. The second degree polynomial

$$\begin{aligned} I\;(du, dv) \;&=\; \mathbf{x}_u \cdot \mathbf{x}_u\; du^2 + 2\;\mathbf{x}_u \cdot \mathbf{x}_v\; du\; dv + \mathbf{x}_v \cdot \mathbf{x}_v\; dv^2 \\ &=\; E\; du^2 + 2F\; du\; dv + G\; dv^2, \end{aligned} \tag{4.7.}$$

where $du, dv \in \mathbb{R}$, is called the **first fundamental form** of a regular parametric surface $\mathbf{x}(\mathbf{u})$. The coefficients E, F, and G are called the **first fundamental coefficients**.

Definition 4.6. Assuming that the regular parametric surface $\mathbf{x}(\mathbf{u})$ is at least of order 2, the second degree polynomial

$$II\ (du, dv) \ = \ -\mathbf{x}_u \cdot \mathbf{n}_u\ du^2 - (\mathbf{x}_u \cdot \mathbf{n}_v + \mathbf{x}_v \cdot \mathbf{n}_u)\ du\ dv - \mathbf{x}_v \cdot \mathbf{n}_v\ dv^2$$

$$= \ \mathbf{x}_{uu} \cdot \mathbf{n}\ du^2 + 2\ \mathbf{x}_{uv} \cdot \mathbf{n}\ du\ dv + \mathbf{x}_{vv} \cdot \mathbf{n}\ dv^2 \ = \ L\ du^2 + 2M\ du\ dv + N\ dv^2, \quad (4.8.)$$

where $du, dv \in \mathbb{R}$, is called the **second fundamental form** of $\mathbf{x}(\mathbf{u})$. The coefficients L, M, and N are called the **second fundamental coefficients**.

Definition 4.7. The two (real) eigenvalues κ_1 and κ_2 of the matrix

$$-A \ = \ -\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \ = \ \begin{pmatrix} L & M \\ M & N \end{pmatrix} \begin{pmatrix} E & F \\ F & G \end{pmatrix}^{-1}, \qquad (4.9.)$$

where

$$a_{1,1} \ = \ \frac{MF - LG}{EG - F^2}, \qquad a_{1,2} \ = \ \frac{LF - ME}{EG - F^2},$$
$$a_{2,1} \ = \ \frac{NF - MG}{EG - F^2}, \qquad a_{2,2} \ = \ \frac{MF - NE}{EG - F^2},$$

of a regular surface of class of at least 2 at a point $\mathbf{x}_0$ are called **principal curvatures** of the regular parametric surface at $\mathbf{x}_0$. The associated eigenvectors determine the **principal curvature directions**. Therefore, the principal curvatures are the (real) roots of the characteristic polynomial of $-A$, the quadratic polynomial

$$\kappa^2 + (a_{1,1} + a_{2,2})\ \kappa + a_{1,1}a_{2,2} - a_{1,2}a_{2,1}. \qquad (4.10.)$$

Remark 4.1. The equations for the matrix elements $a_{i,j}$ in equation (4.9.) are known as the Gauss-Weingarten equations (or the Gauss-Weingarten map).

Remark 4.2. It is shown in [Spivak '70] that the eigenvalues of the matrix $-A$ in equation (4.9.) are always real, and the associated eigenvectors are orthogonal to each other.

Definition 4.8. The average H of the two principal curvatures κ_1 and κ_2 is called the **mean curvature**, the product K is called the **Gaussian curvature** of the regular parametric surface $\mathbf{x}(\mathbf{u})$ at $\mathbf{x}_0$,

$$H \;=\; \frac{1}{2}\,(\kappa_1 + \kappa_2), \qquad\qquad K \;=\; \kappa_1\kappa_2. \tag{4.11.}$$

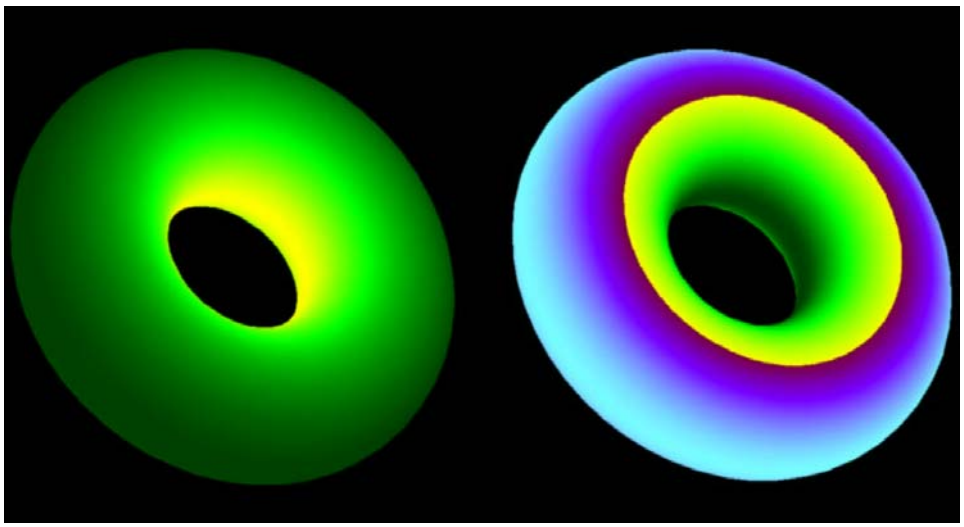

Fig.4.1. Texture map of mean and Gaussian curvature onto a torus,
$\big(\ (2+\cos u)\cos v, (2+\cos u)\sin v, \sin u\ \big)^T,\ u, v \in [0, 2\pi]$;
green/yellow representing negative values,
magenta/blue representing positive values.

4.2. Curvature approximation for triangulated two-dimensional surfaces

The graph of a bivariate function $f(x, y)$, f in class C^m, $m \geq 2$, mapping an open set $U \subset I\!R^2$ into $I\!R$, can be interpreted as a regular parametric two-dimensional surface in three-dimensional space using the parametrization $x(u, v) = u$, $y(u, v) = v$, and $z(u, v) = f(u, v)$,

$$\mathbf{x}\,(\mathbf{u}) \;=\; \big(\; u, v, f(u,v)\;\big)^T, \qquad (u,v) \in D \;\subset\; I\!R^2. \tag{4.12.}$$

For this particular surface, one easily derives the formulae

$$\mathbf{x}_u \;=\; (1, 0, f_u)^T, \qquad \mathbf{x}_v \;=\; (0, 1, f_v)^T,$$

$$\mathbf{x}_{uu} \;=\; (0, 0, f_{uu})^T, \qquad \mathbf{x}_{uv} \;=\; (0, 0, f_{uv})^T, \qquad \mathbf{x}_{vv} \;=\; (0, 0, f_{vv})^T, \qquad \text{and}$$

$$\mathbf{n}\,(\mathbf{u}) \;=\; \frac{\mathbf{x}_u \times \mathbf{x}_v}{||\mathbf{x}_u \times \mathbf{x}_v||} \;=\; \frac{(-f_u, -f_v, 1)^T}{\sqrt{1 + {f_u}^2 + {f_v}^2}}. \tag{4.13.}$$

The first and second fundamental coefficients are therefore given by

$$E \;=\; 1 + {f_u}^2, \qquad F \;=\; f_u f_v \qquad G \;=\; 1 + {f_v}^2,$$

$$L \;=\; \frac{f_{uu}}{\sqrt{1 + {f_u}^2 + {f_v}^2}}, \qquad M \;=\; \frac{f_{uv}}{\sqrt{1 + {f_u}^2 + {f_v}^2}}, \qquad \text{and} \qquad N \;=\; \frac{f_{vv}}{\sqrt{1 + {f_u}^2 + {f_v}^2}}. \tag{4.14.}$$

The Gauss-Weingarten map for this particular surface is given by

$$-A \;=\; -\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \;=\; \frac{1}{l}\begin{pmatrix} f_{uu} & f_{uv} \\ f_{uv} & f_{vv} \end{pmatrix}\begin{pmatrix} 1 + {f_u}^2 & f_u f_v \\ f_u f_v & 1 + {f_v}^2 \end{pmatrix}^{-1}, \tag{4.15.}$$

where $l = \sqrt{1 + {f_u}^2 + {f_v}^2}$.

Theorem 4.1. *Each regular parametric two-dimensional surface* $\mathbf{x}(\mathbf{u})$ *of class* m, $m \geq 2$, *can locally be represented in the explicit form* $z = z(x, y)$ *which is at least* C^2. *Choosing a surface point* $\mathbf{x}_0$ *as origin of a local coordinate system and the z-axis in the same direction as the surface normal* $\mathbf{n}_0$ *at* $\mathbf{x}_0$ *(thus choosing the tangent plane at* $\mathbf{x}_0$ *as the xy-plane), the Taylor series for* z *considering only the terms up to degree* 2 *is given by*

$$z\ (x, y)\ =\ \frac{1}{2}\ \left(c_{2,0}x^2 + 2c_{1,1}xy + c_{0,2}y^2\right), \tag{4.16.}$$

choosing any 2 *unit vectors in the xy-plane determining a right-handed orthonormal coordinate system. Rotating these* 2 *unit vectors appropriately yields the equation of the so-called* **osculating paraboloid** *at* $\mathbf{x}_0$,

$$z\ (x, y)\ =\ \frac{1}{2}\ \left(c^*_{2,0}x^2 + c^*_{0,2}y^2\right)$$

such that the two principal curvatures at $\mathbf{x}_0$ *coincide with the coefficients of this paraboloid,* $\kappa_1 = c^*_{2,0}$ *and* $\kappa_2 = c^*_{0,2}$.

Proof. See [Strubecker '58,'59] or [Struik '61].

The principal curvature approximation method to be introduced is based on bivariate polynomials. It is essential to prove a certain property of such functions before describing the approximation technique. Given an origin in the plane, the graph of a bivariate polynomial f consisting of all the points in the set $\left\{\left(\ x,\ y,\ f\ (x, y)\ \right)^T \mid x, y \in I\!R\right\}$ is independent of the choice of the orientation of the two unit vectors determining an orthonormal coordinate system for the plane. This fact implies that

the principal curvatures of the graph, a two-dimensional surface, are independent of the two unit vectors as well.

Lemma 4.1. *The equation*

$$\sum_{k=0}^{i} (-1)^k \binom{i}{k} \left(x \cos^2\alpha + y \sin\alpha \cos\alpha\right)^{i-k} \left(-x \sin^2\alpha + y \sin\alpha \cos\alpha\right)^k = x^i \quad (4.17.)$$

holds for all $x,\ y,\ \alpha \in \mathbb{R}$ *and* $i \geq 0$.

Proof. It is easy to show that equation (4.17.) is valid for $i = 0$:

$$1 = x^0.$$

The induction hypothesis is made that equation (4.17.) is true for $i - 1$. Thereby one proves that

$$\sum_{k=0}^{i} (-1)^k \binom{i}{k} \left(x \cos^2\alpha + y \sin\alpha \cos\alpha\right)^{i-k} \left(-x \sin^2\alpha + y \sin\alpha \cos\alpha\right)^k$$

$$= \Big(\left(x \cos^2\alpha + y \sin\alpha \cos\alpha\right) - \left(-x \sin^2\alpha + y \sin\alpha \cos\alpha\right)\Big)$$

$$\sum_{k=0}^{i-1} (-1)^k \binom{i-1}{k} \left(x \cos^2\alpha + y \sin\alpha \cos\alpha\right)^{i-1-k} \left(-x \sin^2\alpha + y \sin\alpha \cos\alpha\right)^k$$

$$= x\ (\cos^2\alpha + \sin^2\alpha)\ x^{i-1} = x\ x^{i-1} = x^i.$$

q.e.d.

Lemma 4.2. *The equation*

$$\sum_{l=0}^{j} \binom{j}{l} \left(x \sin\alpha \cos\alpha + y \sin^2\alpha\right)^{j-l} \left(-x \sin\alpha \cos\alpha + y \cos^2\alpha\right)^l = y^j \qquad (4.18.)$$

holds for all $x,\ y,\ \alpha \in \mathbb{R}$ *and* $j \geq 0$.

Proof. Equation (4.18.) holds for $j = 0$:

$$1 = y^0.$$

Using the induction hypothesis that equation (4.18.) is true for $j - 1$, one proves that

$$\begin{aligned}
&\sum_{l=0}^{j} \binom{j}{l} \left(x \sin\alpha \cos\alpha + y \sin^2\alpha\right)^{j-l} \left(-x \sin\alpha \cos\alpha + y \cos^2\alpha\right)^{l} \\
&\quad = \left((x \sin\alpha \cos\alpha + y \sin^2\alpha) + (-x \sin\alpha \cos\alpha + y \cos^2\alpha)\right) \\
&\sum_{l=0}^{j-1} \binom{j-1}{l} \left(x \sin\alpha \cos\alpha + y \sin^2\alpha\right)^{j-1-l} \left(-x \sin\alpha \cos\alpha + y \cos^2\alpha\right)^{l} \\
&\quad = y \ (\sin^2\alpha + \cos^2\alpha) \ y^{j-1} = y \ y^{j-1} = y^j.
\end{aligned}$$

q.e.d.

Lemma 4.1. and Lemma 4.2. are needed to prove the following theorem.

Theorem 4.2. *Let f be the bivariate polynomial*

$$f\ (x, y) = \sum_{\substack{i+j \le n \\ i,j \ge 0}} c_{i,j}\ x^i\ y^j, \tag{4.19.}$$

where a point in the plane has coordinates x and y with respect to a coordinate system given by an origin $\mathbf{o}$ and two orthonormal basis vectors $\mathbf{d}_1$ and $\mathbf{d}_2$; rotating $\mathbf{d}_1$ and $\mathbf{d}_2$ around the origin $\mathbf{o}$ changes the representation of the bivariate polynomial, but not its graph.

Proof. Let $\mathbf{d}_1$ and $\mathbf{d}_2$ be two unit vectors determining a first orthonormal coordinate system together with the origin $\mathbf{o}$, and let $\overline{\mathbf{d}_1}$ and $\overline{\mathbf{d}_2}$ be a second pair of unit vectors obtained by rotating $\mathbf{d}_1$ and $\mathbf{d}_2$ by an angle α around $\mathbf{o}$. A point in the

plane may have coordinates $(x, y)^T$ with respect to the first coordinate system and coordinates

$$\begin{pmatrix} \overline{x} \\ \overline{y} \end{pmatrix} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.20.}$$

with respect to the second coordinate system. Assuming (4.19.) is the representation of the polynomial f with respect to the first coordinate system, f can be rewritten using the inverse map of (4.20.):

$$f\ (x = \overline{x}\cos\alpha - \overline{y}\sin\alpha,\ y = \overline{x}\sin\alpha + \overline{y}\cos\alpha)$$

$$= \sum_{\substack{i+j\leq n \\ i,j\geq 0}} c_{i,j}\ (\overline{x}\cos\alpha - \overline{y}\sin\alpha)^i\ (\overline{x}\sin\alpha + \overline{y}\cos\alpha)^j. \tag{4.21.}$$

Evaluating f at the point $(\overline{x}, \overline{y})^T = (x\cos\alpha + y\sin\alpha,\ -x\sin\alpha + y\cos\alpha)^T$, considering the binomial theorem, Lemma 4.1., and Lemma 4.2., one derives the equations

$$f\ (\overline{x} = x\cos\alpha + y\sin\alpha,\ \overline{y} = -x\sin\alpha + y\cos\alpha)$$

$$= \sum_{\substack{i+j\leq n \\ i,j\geq 0}} c_{i,j}\ \big(\cos\alpha\ (x\cos\alpha + y\sin\alpha) - \sin\alpha\ (-x\sin\alpha + y\cos\alpha)\big)^i$$

$$\big(\sin\alpha\ (x\cos\alpha + y\sin\alpha) + \cos\alpha\ (-x\sin\alpha + y\cos\alpha)\big)^j$$

$$= \sum_{\substack{i+j\leq n \\ i,j\geq 0}} c_{i,j}\ \Big(\sum_{k=0}^{i} (-1)^k \binom{i}{k} \big(\cos\alpha(x\cos\alpha + y\sin\alpha)\big)^{i-k} \big(\sin\alpha(-x\sin\alpha + y\cos\alpha)\big)^k$$

$$\sum_{l=0}^{j} \binom{j}{l}\ \big(\sin\alpha\ (x\cos\alpha + y\sin\alpha)\big)^{j-l}\ \big(\cos\alpha\ (-x\sin\alpha + y\cos\alpha)\big)^l\ \Big)$$

$$= \sum_{\substack{i+j\leq n \\ i,j\geq 0}} c_{i,j} \left(\sum_{k=0}^{i} (-1)^k \binom{i}{k} \left(x\cos^2\alpha + y\sin\alpha\cos\alpha\right)^{i-k} \left(-x\sin^2\alpha + y\sin\alpha\cos\alpha\right)^{k} \right.$$

$$\left. \sum_{l=0}^{j} \binom{j}{l} \left(x\sin\alpha\cos\alpha + y\sin^2\alpha\right)^{j-l} \left(-x\sin\alpha\cos\alpha + y\cos^2\alpha\right)^{l} \right)$$

$$= \sum_{\substack{i+j\leq n \\ i,j\geq 0}} c_{i,j}\ x^i\ y^j = f\ (x,y)$$

proving the theorem.

q.e.d.

The curvature approximation method is based on a localization of a two-dimensional triangulation. The local neighborhood around a point $\mathbf{x}_i$ is its platelet.

Definition 4.9. Given a two-dimensional triangulation in two- or three-dimensional space, the **platelet** $\mathcal{P}_i$ associated with a point $\mathbf{x}_i$ in the triangulation is the set of all triangles (determined by the index-triples (j_1, j_2, j_3) specifying their vertices) sharing $\mathbf{x}_i$ as a common vertex,

$$\mathcal{P}_i \ = \ \bigcup \left\{\ (j_1, j_2, j_3) \mid i = j_1 \vee i = j_2 \vee i = j_3\ \right\}. \tag{4.22.}$$

The vertices constituting $\mathcal{P}_i$ are referred to as **platelet points**.

In order to approximate the principal curvatures at a point $\mathbf{x}_i$ in a two-dimensional triangulation a bivariate polynomial is constructed for a certain neighborhood around this point. Considering the facts that a two-dimensional surface can locally be represented explicitly (Theorem 4.1.) and that the graph of a bivariate polynomial is independent of the orientation of the two unit vectors determining an

orthonormal coordinate system for the plane (Theorem 4.2.), the following sequence of computations is proposed.

(i) Determine the platelet points associated with $\mathbf{x}_i$.

(ii) Compute the plane P passing through $\mathbf{x}_i$ and having $\mathbf{n}_i$ (the normal at $\mathbf{x}_i$) as its normal.

(iii) Define an orthonormal coordinate system in P with $\mathbf{x}_i$ as its origin and two arbitrary unit vectors in P.

(iv) Compute the distances of all platelet points from the plane P.

(v) Project all platelet points onto the plane, P and represent their projections with respect to the local coordinate system in P.

(vi) Interpret the projections in P as abscissae values and the distances of the original platelet points from P as ordinate values.

(vii) Construct a bivariate polynomial f approximating these ordinate values.

(viii) Compute the principal curvatures of f's graph at $\mathbf{x}_i$.

All steps needing further explaining are discussed in greater detail. Let $\left\{\mathbf{y}_j = (x_j, y_j, z_j)^T \mid j = 0...n_i\right\}$ be the set of all platelet points associated with the point $\mathbf{x}_i$ such that $\mathbf{y}_0 = \mathbf{x}_i$, and let $\mathbf{n} = (n^x, n^y, n^z)^T$ be the outward unit normal vector at $\mathbf{y}_0$. The implicit equation for the plane P is then given by

$$\begin{aligned}\mathbf{n} \cdot (\mathbf{x} - \mathbf{y}_0) &= n^x(x - x_0) + n^y(y - y_0) + n^z(z - z_0) \\ &= n^x x + n^y y + n^z z - (n^x x_0 + n^y y_0 + n^z z_0)\end{aligned}$$

$$= \; Ax + By + Cz + D \; = \; 0. \tag{4.23.}$$

Depending on the outward unit normal vector $\mathbf{n}$ one chooses a vector $\mathbf{a}$ perpendicular to $\mathbf{n}$ ($\mathbf{a} \cdot \mathbf{n} = 0$) among the possibilities

$$\mathbf{a} \; = \; \begin{cases} \frac{1}{n^x} \left(-(n^y + n^z), n^x, n^x\right)^T, & n^x \neq 0, \\ \frac{1}{n^y} \left(n^y, -(n^x + n^z), n^y\right)^T, & n^y \neq 0, \\ \frac{1}{n^z} \left(n^z, n^z, -(n^x + n^y)\right)^T, & n^z \neq 0, \end{cases}$$

in order to obtain the first unit basis vector $\mathbf{b}_1$,

$$\mathbf{b}_1 \; = \; \frac{\mathbf{a}}{||\mathbf{a}||}, \qquad\qquad ||\mathbf{a}|| \; = \; \sqrt{(\mathbf{a} \cdot \mathbf{a})}.$$

The second unit basis vector $\mathbf{b}_2$ is defined as the cross product of $\mathbf{n}$ and $\mathbf{b}_1$,

$$\mathbf{b}_2 \; = \; \mathbf{n} \times \mathbf{b}_1.$$

The perpendicular signed distances d_j, $j = 0...n_i$, of all platelet points $\mathbf{y}_j$ from the plane P are

$$d_j \; = \; \text{dist}\,(\mathbf{y}_j, P) \; = \; \frac{Ax_j + By_j + Cz_j + D}{\sqrt{A^2 + B^2 + C^2}} \; = \; Ax_j + By_j + Cz_j + D. \tag{4.24.}$$

Projecting all platelet points $\mathbf{y}_j$ onto P yields the points $\mathbf{y}_j^P$,

$$\mathbf{y}_j^P \; = \; \mathbf{y}_j - d_j \; \mathbf{n}. \tag{4.25.}$$

Considering $\mathbf{y}_0$ as the origin and $\mathbf{b}_1$ and $\mathbf{b}_2$ as the two unit basis vectors of a local two-dimensional orthonormal coordinate system for the plane P, each point $\mathbf{y}_j^P$ in P can be expressed in terms of that coordinate system. Therefore, one computes the difference vectors

$$\mathbf{d}_j = \mathbf{y}_j^P - \mathbf{y}_0, \qquad\qquad j = 0...n_i,$$

and expresses them as linear combinations of the two unit basis vectors $\mathbf{b}_1$ and $\mathbf{b}_2$ in P. Each difference vector $\mathbf{d}_j$ can be represented in the form

$$\mathbf{d}_j \;=\; (\mathbf{d}_j \cdot \mathbf{b}_1)\;\mathbf{b}_1 + (\mathbf{d}_j \cdot \mathbf{b}_2)\;\mathbf{b}_2, \tag{4.26.}$$

defining the local coordinates u_j and v_j of the point $\mathbf{y}_j^P$ in terms of the local coordinate system:

$$\big(u_j, v_j\big)^T \;=\; \big(\mathbf{d}_j \cdot \mathbf{b}_1, \mathbf{d}_j \cdot \mathbf{b}_2\big)^T. \tag{4.27.}$$

Interpreting the local coordinates u_j and v_j as abscissae values and the signed distances d_j as ordinate values (in direction of the normal $\mathbf{n}$), a polynomial $f(u, v)$ of degree two (see Theorem 4.1.) is constructed approximating these ordinate values. Forcing the polynomial f to satisfy $f(0,0) = f_u(0,0) = f_v(0,0) = 0$, the constraints

$$f\;(u_j, v_j) \;=\; \frac{1}{2}\left(c_{2,0}{u_j}^2 + 2c_{1,1}u_j v_j + c_{0,2}{v_j}^2\right) \;=\; d_j, \qquad j = 1...n_i,$$

remain. Written in matrix representation these constraints are

$$\begin{pmatrix} {u_1}^2 & 2u_1v_1 & {v_1}^2 \\ \vdots & \vdots & \vdots \\ {u_{n_i}}^2 & 2u_{n_i}v_{n_i} & {v_{n_i}}^2 \end{pmatrix} \begin{pmatrix} c_{2,0} \\ c_{1,1} \\ c_{0,2} \end{pmatrix} \;=\; U\;\mathbf{c} \;=\; \mathbf{d} \;=\; \begin{pmatrix} d_1 \\ \vdots \\ d_{n_i} \end{pmatrix}. \tag{4.28.}$$

This overdetermined system of linear equations is solved using a least squares approach (see [Davis '75]). The resulting normal equations are

$$U^T\;U\;\mathbf{c} \;=\; U^T\;\mathbf{d}. \tag{4.29.}$$

Provided the determinant of U^TU does not vanish this $3 \cdot 3-$system of linear equations can immediately be solved using Cramer's rule.

Theorem 4.3. *The principal curvatures κ_1 and κ_2 of the graph $\left(u, v, f(u,v) \right)^T \subset \mathbb{R}^3$, $u, v \in \mathbb{R}$, of the bivariate polynomial*

$$f\ (u,v) \ = \ \frac{1}{2} \left(c_{2,0} u^2 + 2 c_{1,1} uv + c_{0,2} v^2 \right) \tag{4.30.}$$

at the point $\left(0, 0, f(0,0) \right)^T$ are given by the two real roots of the quadratic equation

$$\kappa^2 - (c_{2,0} + c_{0,2})\ \kappa + c_{2,0} c_{0,2} - {c_{1,1}}^2 \ = \ 0. \tag{4.31.}$$

Proof. According to Definition 4.7. and equation (4.15.), the principal curvatures of f's graph are the eigenvalues of the matrix

$$-A \ = \ \frac{1}{l} \begin{pmatrix} f_{uu} & f_{uv} \\ f_{uv} & f_{vv} \end{pmatrix} \begin{pmatrix} 1 + {f_u}^2 & f_u f_v \\ f_u f_v & 1 + {f_v}^2 \end{pmatrix}^{-1},$$

where $l = \sqrt{1 + {f_u}^2 + {f_v}^2}$. Evaluating $-A$ for $u = v = 0$, one obtains the matrix

$$-A \ = \ \begin{pmatrix} c_{2,0} & c_{1,1} \\ c_{1,1} & c_{0,2} \end{pmatrix},$$

having the characteristic polynomial in (4.31.).

q.e.d.

Solving the normal equations (4.29.) and determining the roots of the characteristic polynomial in (4.31.), one finally obtains the desired approximations for the principal curvatures at the point $\mathbf{x}_i$.

The above construction is illustrated in Figure 4.2. Shown are the platelet points around the point $\mathbf{x}_i$, the tangent plane P, its local orthonormal coordinate

system (origin $\mathbf{x}_i$ and basis vectors $\mathbf{b}_1$ and $\mathbf{b}_2$), and the projections of the platelet points ($\mathbf{y}_j^P$) onto P.

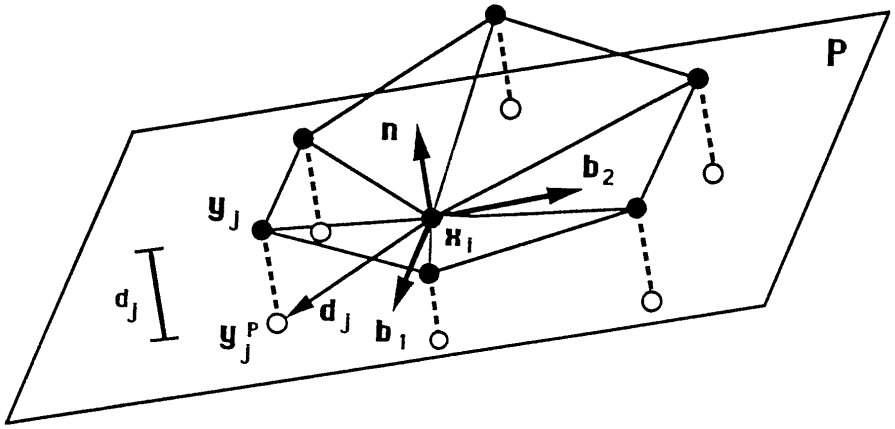

Fig. 4.2. Construction of a bivariate polynomial for platelet points in a two-dimensional triangulation.

The presented technique for principal curvature approximation is tested for graphs of several bivariate functions. The exact principal curvatures ${\kappa_1}^{ex}$ and ${\kappa_2}^{ex}$ are compared with the approximated principal curvatures ${\kappa_1}^{app}$ and ${\kappa_2}^{app}$; the exact mean curvature $H^{ex} = \frac{1}{2}\ ({\kappa_1}^{ex} + {\kappa_2}^{ex})$ is compared with the average of the approximated principal curvatures $H^{app} = \frac{1}{2}\ ({\kappa_1}^{app} + {\kappa_2}^{app})$ and the exact Gaussian curvature $K^{ex} = {\kappa_1}^{ex}{\kappa_2}^{ex}$ with the product of the approximated principal curvatures $K^{app} = {\kappa_1}^{app}{\kappa_2}^{app}$.

All bivariate test functions $f(x, y)$ are defined over $[-1, 1] \times [-1, 1]$ and evaluated on a $51 \cdot 51-$grid with equidistant spacing,

$$(x_i, y_j)^T \;=\; \Big(-1 + \frac{i}{25}, -1 + \frac{j}{25} \Big)^T, \qquad i, j = 0...50,$$

determining a finite set of three-dimensional points on their graphs,

$$\Big\{ \big(\; x_i, y_j, f(x_i, y_j) \; \big)^T \mid i, j = 0...50 \Big\}.$$

The triangulation of a function's graph is obtained by splitting each quadrilateral specified by its index quadruple

$$\big(\; (i, j), (i + 1, j), (i + 1, j + 1), (i, j + 1) \; \big)$$

into the two triangles $T^1_{i,j}$ and $T^2_{i,j}$ identified by their index triples,

$$T^1_{i,j} \;=\; \big(\; (i, j), (i{+}1, j), (i{+}1, j{+}1) \; \big) \qquad \text{and} \qquad T^2_{i,j} \;=\; \big(\; (i, j), (i{+}1, j{+}1), (i, j{+}1) \; \big).$$

The root-mean-square error (RMS error) is a common error measure and is computed for each test example and curvature type. The RMS error is defined as

$$\sqrt{\frac{1}{n} \sum_{i=0}^{n-1} ({f_i}^{ex} - {f_i}^{app})^2} \tag{4.32.}$$

where n is the total number of exact (or approximated) values ${f_i}^{ex}$ (${f_i}^{app}$). Here, n equals $51 \cdot 51$, depending on the curvature type approximated ${f_i}^{ex}$ can represent the exact values for ${\kappa_1}^{ex}$, ${\kappa_2}^{ex}$, H^{ex} or K^{ex}, and ${f_i}^{app}$ can represent the approximated values for ${\kappa_1}^{app}$, ${\kappa_2}^{app}$, H^{app} or K^{app}, respectively. Table 4.1. summarizes the test results for the approximation of the principal curvatures, the mean and the Gaussian curvature.

Tab. 4.1. RMS errors of curvature approximation for graphs of bivariate functions.

Function	κ_1	κ_2	H	K
1. Plane: $.2\ (x+y)$.	0	0	0	0
2. Cylinder: $\sqrt{2-x^2}$.	.000291	.000035	.000132	.000025
3. Sphere: $\sqrt{4-(x^2+y^2)}$.	.000159	.000046	.000080	.000080
4. Paraboloid: $.4\ (x^2+y^2)$.	.003073	.001342	.001358	.001684
5. Hyperboloid: $.4\ (x^2-y^2)$.	.002058	.002058	.001057	.001767
6. Monkey saddle: $.2\ (x^3-3xy^2)$.	.004483	.004483	.001591	.007247
7. Cubic polynomial: $.15\ (x^3+2x^2y-xy+2y^2)$.	.002258	.003598	.001665	.002242
8. Exponential function: $e^{-\frac{1}{2}(x^2+y^2)}$.	.001757	.005546	.002722	.002602
9. Trigonometric function: $.1\ \big(\cos(\pi x)+\cos(\pi y)\big)$.	.002998	.002821	.001013	.003541

In the following figures, the four particular curvatures used in Table 4.1. are mapped as textures onto the hyperboloid (function 5), the graph of the cubic polynomial (function 7) and the graph of the trigonometric function (function 9). Pairs of consecutive figures show the exact (upper figure) and the approximated curvatures (lower figure). The principal curvature κ_1 is visualized in the upper-left, κ_2 in the upper-right, the mean curvature H in the lower-left and the Gaussian curvature K in the lower-right corner of each figure. Figures 4.3. and 4.4. show the exact and approximated curvature values for function 5, Figures 4.5. and 4.6. for function 7, and Figures 4.7. and 4.8. for function 9.

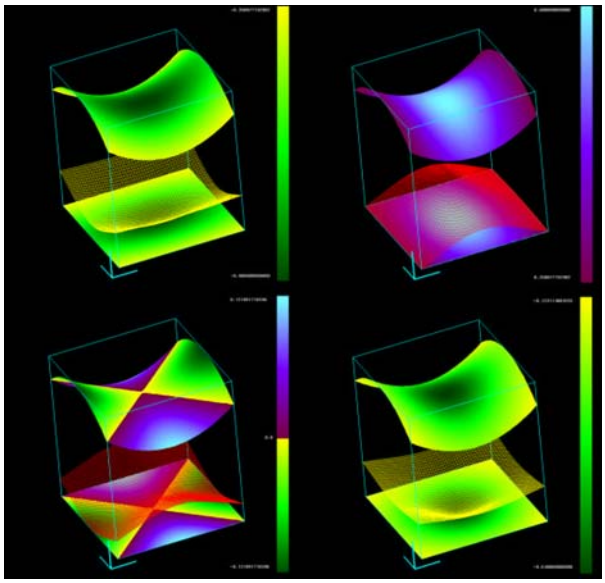

Fig. 4.3. Exact curvatures ${\kappa_1}^{ex}$, ${\kappa_2}^{ex}$, H^{ex}, and K^{ex} on the graph of $f(x,y) = .4\ (x^2 - y^2),\ x, y \in [-1, 1]$.

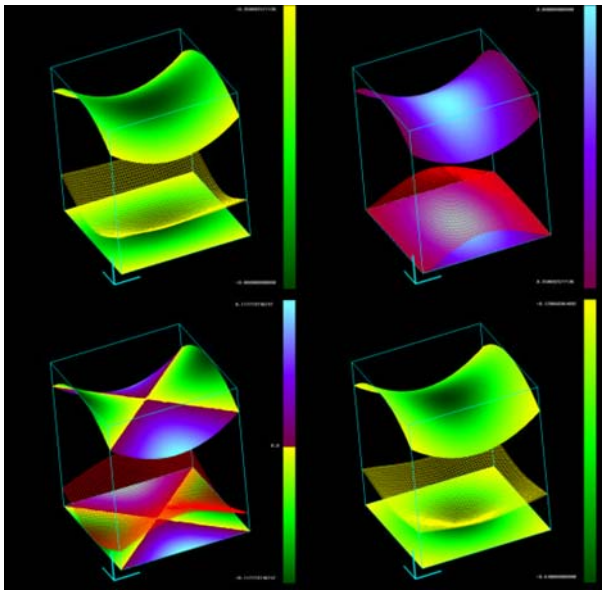

Fig. 4.4. Approximated curvatures ${\kappa_1}^{app}$, ${\kappa_2}^{app}$, H^{app}, and K^{app} on the graph of $f(x,y) = .4\ (x^2 - y^2),\ x, y \in [-1, 1]$.

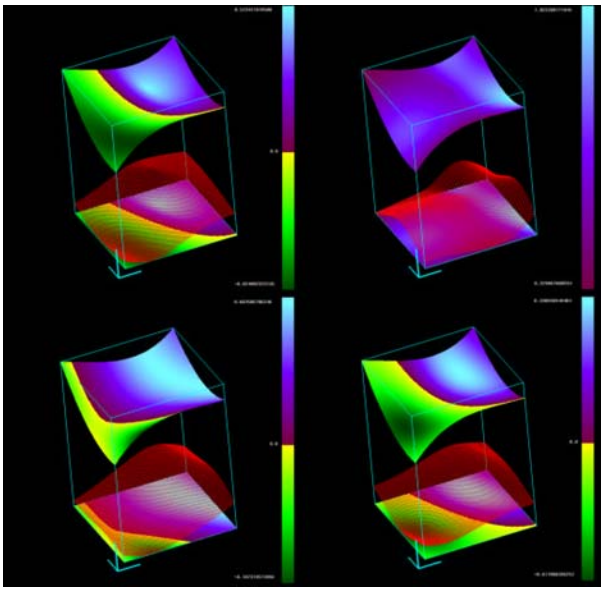

Fig. 4.5. Exact curvatures ${\kappa_1}^{ex}$, ${\kappa_2}^{ex}$, H^{ex}, and K^{ex}
on the graph of $f(x,y) = .15\ (x^3 + 2x^2y - xy + 2y^2),\ x, y \in [-1, 1]$.

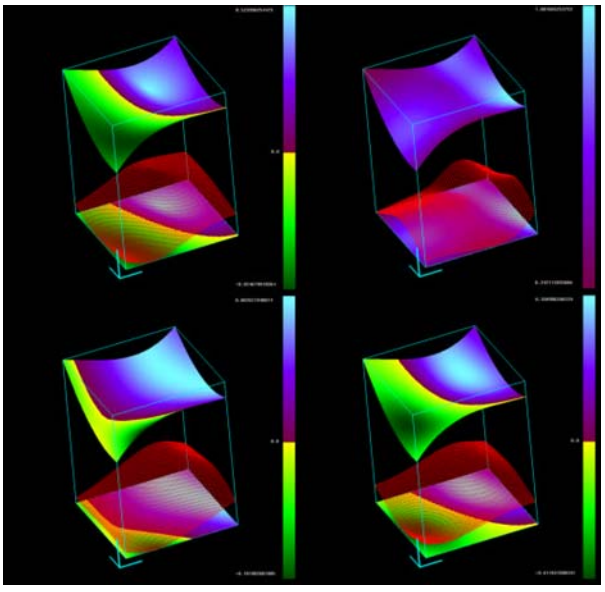

Fig. 4.6. Approximated curvatures ${\kappa_1}^{app}$, ${\kappa_2}^{app}$, H^{app}, and K^{app}
on the graph of $f(x,y) = .15\ (x^3 + 2x^2y - xy + 2y^2),\ x, y \in [-1, 1]$.

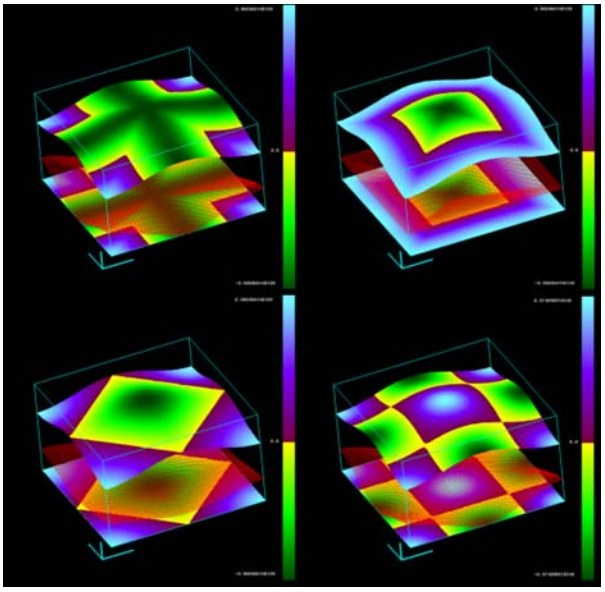

Fig. 4.7. Exact curvatures ${\kappa_1}^{ex}$, ${\kappa_2}^{ex}$, H^{ex}, and K^{ex} on the graph of $f(x,y) = .1\,\big(\cos(\pi x) + \cos(\pi y)\big),\ x, y \in [-1, 1]$.

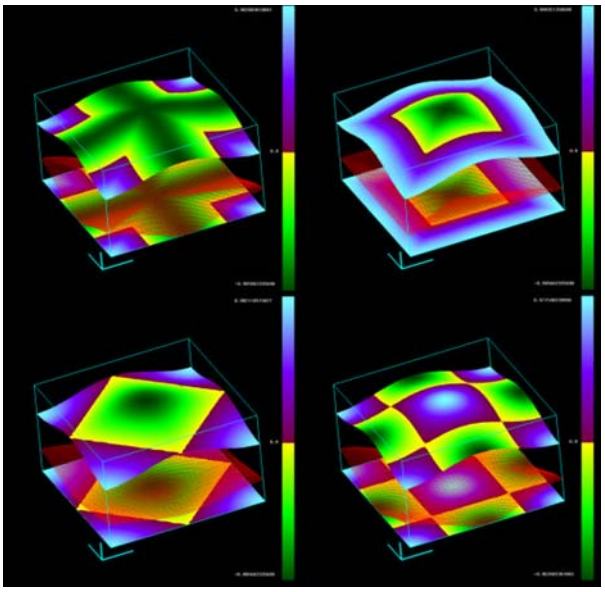

Fig. 4.8. Approximated curvatures ${\kappa_1}^{app}$, ${\kappa_2}^{app}$, H^{app}, and K^{app} on the graph of $f(x,y) = .1\,\big(\cos(\pi x) + \cos(\pi y)\big),\ x, y \in [-1, 1]$.

4.3. Curvature approximation for triangulated three-dimensional graphs of trivariate functions

The graph of a trivariate function $f(x,y,z)$, f in class C^m, $m \geq 2$, mapping an open set $U \subset I\!R^3$ into $I\!R$ can be interpreted as a regular parametric three-dimensional surface in four-dimensional space (see Definition 4.1.) using the parametrization $x(u,v,w) = u$, $y(u,v,w) = v$, $z(u,v,w) = w$, and $W(u,v,w) = f(u,v,w)$,

$$\mathbf{x}\,(\mathbf{u}) \;=\; \big(\; u, v, w, f(u,v,w)\;\big)^T, \qquad (u,v,w) \in D \;\subset\; I\!R^3. \tag{4.33.}$$

For this particular hypersurface, one easily derives the formulae

$$\mathbf{x}_u \;=\; (1,0,0,f_u)^T, \qquad \mathbf{x}_v \;=\; (0,1,0,f_v)^T, \qquad \mathbf{x}_w \;=\; (0,0,1,f_w)^T,$$

$$\mathbf{x}_{uu} \;=\; (0,0,0,f_{uu})^T, \qquad \mathbf{x}_{uv} \;=\; (0,0,0,f_{uv})^T, \qquad \mathbf{x}_{uw} \;=\; (0,0,0,f_{uw})^T,$$

$$\mathbf{x}_{vv} \;=\; (0,0,0,f_{vv})^T, \qquad \mathbf{x}_{vw} \;=\; (0,0,0,f_{vw})^T, \qquad \mathbf{x}_{ww} \;=\; (0,0,0,f_{ww})^T, \qquad \text{and}$$

$$\mathbf{n}\,(\mathbf{u}) \;=\; \frac{\text{cross product } (\mathbf{x}_u, \mathbf{x}_v, \mathbf{x}_w)}{||\text{ cross product } (\mathbf{x}_u, \mathbf{x}_v, \mathbf{x}_w)\;||} \;=\; \frac{(-f_u, -f_v, -f_w, 1)^T}{\sqrt{1 + {f_u}^2 + {f_v}^2 + {f_w}^2}} \tag{4.34.}$$

(for the n-dimensional cross product see [Weld '90]).

Definition 4.10. The three-dimensional **tangent space** at a point $\mathbf{x}_0 = \mathbf{x}(\mathbf{u}_0)$ on a regular parametric three-dimensional surface in four-dimensional space is defined as the set of all points $\mathbf{y}$ in $I\!R^4$ satisfying the equation

$$\mathbf{y} \;=\; \mathbf{x}_0 + a\mathbf{x}_u(\mathbf{u}_0) + b\mathbf{x}_v(\mathbf{u}_0) + c\mathbf{x}_w(\mathbf{u}_0), \qquad a,b,c \in I\!R. \tag{4.35.}$$

The Gauss-Weingarten map for this special graph interpreted as a three-dimensional hypersurface in four-dimensional space is given by

$$-A \;=\; -\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}$$

$$= \;\frac{1}{l}\begin{pmatrix} f_{uu} & f_{uv} & f_{uw} \\ f_{uv} & f_{vv} & f_{vw} \\ f_{uw} & f_{vw} & f_{ww} \end{pmatrix}\begin{pmatrix} 1+{f_u}^2 & f_u f_v & f_u f_w \\ f_u f_v & 1+{f_v}^2 & f_v f_w \\ f_u f_w & f_v f_w & 1+{f_w}^2 \end{pmatrix}^{-1}, \qquad (4.36.)$$

where $l = \sqrt{1+{f_u}^2+{f_v}^2+{f_w}^2}$.

Definition 4.11. The three (real) eigenvalues κ_1, κ_2, and κ_3 of the matrix $-A$ from equation (4.36.) are called the **principal curvatures** of the three-dimensional graph of the trivariate function $f(x, y, z)$. Therefore, the principal curvatures are the (real) roots of the characteristic polynomial of $-A$, the cubic polynomial

$$\kappa^3+(a_{1,1}+a_{2,2}+a_{3,3})\,\kappa^2+(a_{1,1}a_{2,2}+a_{1,1}a_{3,3}+a_{2,2}a_{3,3}-a_{1,2}a_{2,1}-a_{1,3}a_{3,1}-a_{2,3}a_{3,2})\,\kappa$$

$$+(a_{1,1}a_{2,2}a_{3,3}+a_{1,2}a_{2,3}a_{3,1}+a_{1,3}a_{2,1}a_{3,2}-a_{1,1}a_{2,3}a_{3,2}-a_{1,2}a_{2,1}a_{3,3}-a_{1,3}a_{2,2}a_{3,1}). \qquad (4.37.)$$

The average H of the principal curvatures is called the **mean curvature**, the product K is called the **Gaussian curvature**,

$$H \;=\; \frac{1}{3}\,(\kappa_1+\kappa_2+\kappa_3), \qquad\qquad K \;=\; \kappa_1\kappa_2\kappa_3. \qquad (4.38.)$$

Figure 4.9. shows the mean (left) and the Gaussian curvature (right) in three planes intersecting the three-dimensional domain of a trivariate function using the visualization technique described in chapter 2.3. (slicing). Curvature changes in f's graph can clearly be recognized, giving rise to the use of these particular curvature measures as indicators for the smoothness of trivariate functions.

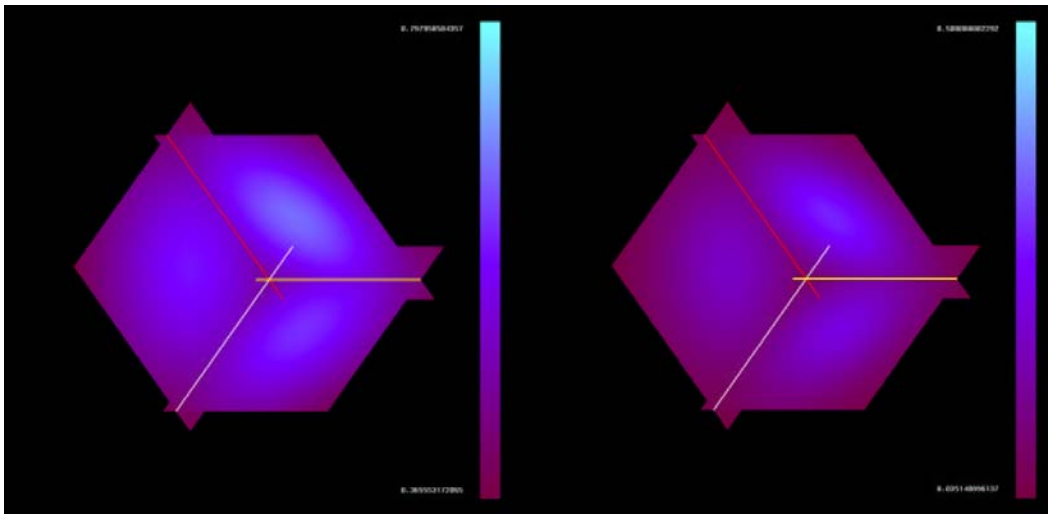

Fig. 4.9. Mean and Gaussian curvature of the graph of $f(x,y,z) = .4\,\left(x^2+y^2+z^2\right),\ x,y,z \in [-1,1]$.

The properties of three-dimensional surfaces stated in the following theorems are needed for the curvature approximation method to be deduced subsequently.

Theorem 4.4. *Each regular parametric three-dimensional surface* $\mathbf{x}(\mathbf{u})$ *of class* m, $m \geq 2$, *can locally be represented in the explicit form* $W = W(x,y,z)$, *where* W *is an at least* C^2 *function. Choosing a surface point* $\mathbf{x}_0$ *as origin of a local coordinate system and the* W*-axis in the same direction as the surface normal* $\mathbf{n}_0$ *at* $\mathbf{x}_0$, *the Taylor series for* W *considering only the terms up to degree* 2 *is given by*

$$W\ (x,y,z)\ =\ \frac{1}{2}\left(c_{2,0,0}x^2 + 2c_{1,1,0}xy + 2c_{1,0,1}xz + c_{0,2,0}y^2 + 2c_{0,1,1}yz + c_{0,0,2}z^2\right), \tag{4.39.}$$

choosing any 3 *unit vectors in the xyz-tangent space determining a right-handed orthonormal coordinate system. Changing the orientation of these* 3 *unit vectors appropriately yields the equation of the so-called* **osculating paraboloid** *at* $\mathbf{x}_0$,

$$W\ (x,y,z)\ =\ \frac{1}{2}\ \left(c^*_{2,0,0}x^2 + c^*_{0,2,0}y^2 + c^*_{0,0,2}z^2\right)$$

such that the three principal curvatures at $\mathbf{x}_0$ *coincide with the coefficients of this paraboloid,* $\kappa_1 = c^*_{2,0,0}$, $\kappa_2 = c^*_{0,2,0}$, *and* $\kappa_3 = c^*_{0,0,2}$.

Proof. See [Strubecker '58,'59] or [Spivak '70].

Theorem 4.5. *Let* f *be the trivariate polynomial*

$$f\ (x,y,z) = \sum_{\substack{i+j+k\leq n \\ i,j,k\geq 0}} c_{i,j,k}\ x^i\ y^j\ z^k, \tag{4.40.}$$

where a point in space has coordinates x, y, *and* z *with respect to a coordinate system given by an origin* $\mathbf{o}$ *and three orthonormal basis vectors* $\mathbf{d}_1$, $\mathbf{d}_2$, *and* $\mathbf{d}_3$; *changing the orientation of the orthonormal basis vectors changes the representation of the trivariate polynomial, but not its graph.*

Proof. Analogous to the proof of Theorem 4.2.

As for the two-dimensional case, the principal curvature approximation technique requires a localization of a three-dimensional triangulation.

Definition 4.12. Given a three-dimensional triangulation (also referred to as a tetrahedrization) in three- or four-dimensional space, the **platelet** $\mathcal{P}_i$ associated with a point $\mathbf{x}_i$ in the triangulation is the set of all tetrahedra (determined by the index-quadruples (j_1, j_2, j_3, j_4) specifying their vertices) sharing $\mathbf{x}_i$ as a common

vertex,

$$\mathcal{P}_i \;=\; \bigcup \{(j_1, j_2, j_3, j_4) \mid i = j_1 \vee i = j_2 \vee i = j_3 \vee i = j_4\}. \tag{4.41.}$$

The vertices constituting $\mathcal{P}_i$ are referred to as **platelet points**.

The sequence of computations for principal curvature approximation in the two-dimensional case, described in chapter 4.2., can easily be extended to the three-dimensional case. The following steps must be executed.

(i) Determine the platelet points associated with $\mathbf{x}_i$.

(ii) Compute the tangent space P passing through $\mathbf{x}_i$ and having $\mathbf{n}_i$ (the normal at $\mathbf{x}_i$) as its normal.

(iii) Define an orthonormal coordinate system in P with $\mathbf{x}_i$ as its origin and three arbitrary unit vectors in P.

(iv) Compute the distances of all platelet points from the tangent space P.

(v) Project all platelet points onto the tangent space P, and represent their projections with respect to the local coordinate system in P.

(vi) Interpret the projections in P as abscissae values and the distances of the original platelet points from P as ordinate values.

(vii) Construct a trivariate polynomial f approximating these ordinate values.

(viii) Compute the principal curvatures of f's graph at $\mathbf{x}_i$.

Some steps are now explained in more detail. Let $\{\ \mathbf{y}_j = (\ x_j, y_j, z_j, W_j\ )^T \mid j = 0...n_i\ \}$ be the set of all platelet points associated with the point $\mathbf{x}_i$ such that $\mathbf{y}_0 = \mathbf{x}_i$, and let $\mathbf{n} = (n^x, n^y, n^z, n^W)^T$ be the outward unit normal vector at $\mathbf{y}_0$. The implicit equation for the tangent space P is given by

$$\begin{aligned}\mathbf{n} \cdot (\mathbf{x} - \mathbf{y}_0) &= n^x(x - x_0) + n^y(y - y_0) + n^z(z - z_0) + n^W(W - W_0) \\ &= n^x x + n^y y + n^z z + n^W W - (n^x x_0 + n^y y_0 + n^z z_0 + n^W W_0) \\ &= Ax + By + Cz + DW + E = 0. \qquad (4.42.)\end{aligned}$$

Clearly, the four vectors

$$\mathbf{n} = \frac{(-f_u, -f_v, -f_w, 1)^T}{\sqrt{1 + {f_u}^2 + {f_v}^2 + {f_w}^2}},$$

$$\mathbf{a}_1 = (1,0,0,0)^T, \quad \mathbf{a}_2 = (0,1,0,0)^T, \quad \text{and} \quad \mathbf{a}_3 = (0,0,1,0)^T$$

are linearly independent and therefore form a basis for $I\!R^4$. Obviously, $\mathbf{a}_1$, $\mathbf{a}_2$, and $\mathbf{a}_3$ are not necessarily perpendicular to the normal $\mathbf{n}$. Using Gram-Schmidt orthogonalization yields an orthonormal basis for $I\!R^4$ consisting of the basis vectors $\mathbf{n}$, $\mathbf{b}_1$, $\mathbf{b}_2$, and $\mathbf{b}_3$, where $\mathbf{b}_1$, $\mathbf{b}_2$, and $\mathbf{b}_3$ are computed as

$$\mathbf{b}_1 = (\mathbf{a}_1 \cdot \mathbf{n})\ \mathbf{n}, \qquad \mathbf{b}_1 = \mathbf{a}_1 - \mathbf{b}_1, \qquad \mathbf{b}_1 = \frac{\mathbf{b}_1}{||\mathbf{b}_1||},$$

$$\mathbf{b}_2 = (\mathbf{a}_2 \cdot \mathbf{n})\ \mathbf{n} + (\mathbf{a}_2 \cdot \mathbf{b}_1)\ \mathbf{b}_1, \qquad \mathbf{b}_2 = \mathbf{a}_2 - \mathbf{b}_2, \qquad \mathbf{b}_2 = \frac{\mathbf{b}_2}{||\mathbf{b}_2||}, \qquad \text{and}$$

$$\mathbf{b}_3 = (\mathbf{a}_3 \cdot \mathbf{n})\ \mathbf{n} + (\mathbf{a}_3 \cdot \mathbf{b}_1)\ \mathbf{b}_1 + (\mathbf{a}_3 \cdot \mathbf{b}_2)\ \mathbf{b}_2, \qquad \mathbf{b}_3 = \mathbf{a}_3 - \mathbf{b}_3, \qquad \mathbf{b}_3 = \frac{\mathbf{b}_3}{||\mathbf{b}_3||},$$

$||\mathbf{b}_i|| = \sqrt{(\mathbf{b}_i \cdot \mathbf{b}_i)}$, $i = 1, 2, 3$.

The perpendicular signed distances d_j, $j = 0...n_i$, of all platelet points $\mathbf{y}_j$ from the tangent space P are

$$d_j \;=\; \text{dist}\,(\mathbf{y}_j, P) \;=\; \frac{Ax_j + By_j + Cz_j + DW_j + E}{\sqrt{A^2 + B^2 + C^2 + D^2}} \;=\; Ax_j + By_j + Cz_j + DW_j + E. \tag{4.43.}$$

Projecting all platelet points $\mathbf{y}_j$ onto P yields the points $\mathbf{y}_j^P$, where

$$\mathbf{y}_j^P \;=\; \mathbf{y}_j - d_j\;\mathbf{n}. \tag{4.44.}$$

Again, $\mathbf{y}_0$ is seen as the origin, and $\mathbf{b}_1$, $\mathbf{b}_2$, and $\mathbf{b}_3$ are regarded as the three unit basis vectors of a local three-dimensional orthonormal coordinate system for the tangent space P. Each point $\mathbf{y}_j^P$ in P is expressed in terms of that coordinate system. Computing the difference vectors $\mathbf{d}_j$ as

$$\mathbf{d}_j = \mathbf{y}_j^P - \mathbf{y}_0, \qquad\qquad j = 0...n_i,$$

and expressing them as linear combinations of the basis vectors $\mathbf{b}_1$, $\mathbf{b}_2$, and $\mathbf{b}_3$ in P, one obtains a new representation for $\mathbf{d}_j$,

$$\mathbf{d}_j \;=\; (\mathbf{d}_j \cdot \mathbf{b}_1)\;\mathbf{b}_1 + (\mathbf{d}_j \cdot \mathbf{b}_2)\;\mathbf{b}_2 + (\mathbf{d}_j \cdot \mathbf{b}_3)\;\mathbf{b}_3, \tag{4.45.}$$

defining the local coordinates u_j, v_j, and w_j of the point $\mathbf{y}_j^P$ in terms of the local coordinate system:

$$\big(u_j, v_j, w_j\big)^T \;=\; \big(\mathbf{d}_j \cdot \mathbf{b}_1, \mathbf{d}_j \cdot \mathbf{b}_2, \mathbf{d}_j \cdot \mathbf{b}_3\big)^T. \tag{4.46.}$$

The local coordinates u_j, v_j, and w_j define the abscissae values and the signed distances d_j the ordinate values (in direction of the normal $\mathbf{n}$) for a polynomial

$f(u,v,w)$ of degree two (see Theorem 4.4.) which is constructed by approximating these ordinate values. Forcing f to satisfy the conditions $f(0,0,0) = f_u(0,0,0) = f_v(0,0,0) = f_w(0,0,0) = 0$ the constraints

$$f\ (u_j, v_j, w_j)\ =$$

$$\frac{1}{2}\ \left(c_{2,0,0}{u_j}^2 + 2c_{1,1,0}u_jv_j + 2c_{1,0,1}u_jw_j + c_{0,2,0}{v_j}^2 + 2c_{0,1,1}v_jw_j + c_{0,0,2}{w_j}^2\right)\ =\ d_j,$$

$j = 1...n_i$, remain. In matrix representation, these constraints are

$$\begin{pmatrix} {u_1}^2 & 2u_1v_1 & 2u_1w_1 & {v_1}^2 & 2v_1w_1 & {w_1}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ {u_{n_i}}^2 & 2u_{n_i}v_{n_i} & 2u_{n_i}w_{n_i} & {v_{n_i}}^2 & 2v_{n_i}w_{n_i} & {w_{n_i}}^2 \end{pmatrix} \begin{pmatrix} c_{2,0,0} \\ c_{1,1,0} \\ c_{1,0,1} \\ c_{0,2,0} \\ c_{0,1,1} \\ c_{0,0,2} \end{pmatrix}$$

$$=\ U\ \mathbf{c}\ =\ \mathbf{d}\ =\ \begin{pmatrix} d_1 \\ \vdots \\ d_{n_i} \end{pmatrix}. \tag{4.47.}$$

Using the least squares approach, the resulting normal equations are

$$U^T\ U\ \mathbf{c}\ =\ U^T\ \mathbf{d}. \tag{4.48.}$$

This $6 \cdot 6-$system of linear equations can easily be solved using Gaussian elimination provided the determinant of U^TU does not vanish.

A theorem in multi-dimensional differential geometry ensures that the three principal curvatures at a point on the graph of a trivariate function are always real.

Theorem 4.6. *The principal curvatures κ_1, κ_2, and κ_3 at any point on the graph $\left(\ u, v, w, f(u,v,w)\ \right)^T\ \subset\ \mathbb{R}^4$, $u, v, w \in \mathbb{R}$, of a trivariate function f of class m,*

$m \geq 2$, are real and are the eigenvalues of the Gauss-Weingarten map associated with its graph at a particular point.

Proof. See [Spivak '70] or [Weld '90].

Theorem 4.7. *The three principal curvatures κ_1, κ_2, and κ_3 of the graph $\big(\ u,\ v,\ w,\ f(u,v,w)\ \big)^T \subset I\!R^4$, $u,v,w \in I\!R$, of the trivariate polynomial*

$$f\ (u,v,w)\ =\ \frac{1}{2}\left(c_{2,0,0}u^2 + 2c_{1,1,0}uv + 2c_{1,0,1}uw + c_{0,2,0}v^2 + 2c_{0,1,1}vw + c_{0,0,2}w^2\right) \tag{4.49.}$$

at the point $\big(\ 0,0,0,f(0,0,0)\ \big)^T$ are given by the three real roots of the cubic equation

$$-\ \kappa^3 \quad +\quad \left(c_{2,0,0} + c_{0,2,0} + c_{0,0,2}\right)\kappa^2$$

$$-\ \left(c_{2,0,0}c_{0,2,0} + c_{2,0,0}c_{0,0,2} + c_{0,2,0}c_{0,0,2} - {c_{1,1,0}}^2 - {c_{1,0,1}}^2 - {c_{0,1,1}}^2\right)\kappa$$

$$+\ \left(c_{2,0,0}c_{0,2,0}c_{0,0,2} + 2c_{1,1,0}c_{1,0,1}c_{0,1,1} - c_{2,0,0}{c_{0,1,1}}^2 - c_{0,2,0}{c_{1,0,1}}^2 - c_{0,0,2}{c_{1,1,0}}^2\right)\ =\ 0. \tag{4.50.}$$

Proof. According to Definition 4.11. and equation (4.36.) the principal curvatures of f's graph are the eigenvalues of the matrix

$$-A\ =\ \frac{1}{l}\begin{pmatrix} f_{uu} & f_{uv} & f_{uw} \\ f_{uv} & f_{vv} & f_{vw} \\ f_{uw} & f_{vw} & f_{ww} \end{pmatrix}\begin{pmatrix} 1+{f_u}^2 & f_uf_v & f_uf_w \\ f_uf_v & 1+{f_v}^2 & f_vf_w \\ f_uf_w & f_vf_w & 1+{f_w}^2 \end{pmatrix}^{-1},$$

where $l = \sqrt{1+{f_u}^2+{f_v}^2+{f_w}^2}$. Evaluating $-A$ for $u=v=w=0$ one obtains the symmetric matrix

$$-A\ =\ \begin{pmatrix} c_{2,0,0} & c_{1,1,0} & c_{1,0,1} \\ c_{1,1,0} & c_{0,2,0} & c_{0,1,1} \\ c_{1,0,1} & c_{0,1,1} & c_{0,0,2} \end{pmatrix},$$

having the characteristic polynomial in (4.50.).

q.e.d.

The roots of the characteristic polynomial in (4.50.) finally determine the approximations for the principal curvatures at the point $\mathbf{x}_i$.

Remark 4.3. It is well known in linear algebra that the eigenvalues of a symmetric matrix are all real (see [Lang '66]). Considering this fact, it is obvious that the three roots of the cubic characteristic polynomial appearing in Theorem 4.7. must also be real since the matrix $-A$ is symmetric.

Remark 4.4. The first root of the cubic polynomial in equation (4.50.) is computed using Newton's method. The other two roots are calculated after factorization of the cubic polynomial.

The principal curvature approximation technique is examined for graphs of six trivariate functions. The exact mean curvature $H^{ex} = \frac{1}{3}\ ({\kappa_1}^{ex} + {\kappa_2}^{ex} + {\kappa_3}^{ex})$ is compared with the average of the approximated principal curvatures $H^{app} = \frac{1}{3}\ ({\kappa_1}^{app} + {\kappa_2}^{app} + {\kappa_3}^{app})$ and the exact Gaussian curvature $K^{ex} = {\kappa_1}^{ex}{\kappa_2}^{ex}{\kappa_3}^{ex}$ with the product of the approximated principal curvatures $K^{app} = {\kappa_1}^{app}{\kappa_2}^{app}{\kappa_3}^{app}$.

All trivariate test functions $f(x, y, z)$ are defined over $[-1, 1] \times [-1, 1] \times [-1, 1]$ and are evaluated on a $26 \cdot 26 \cdot 26-$grid with equidistant spacing,

$$(x_i, y_j, z_k)^T \ = \ \Big(-1 + \frac{i}{12.5}, -1 + \frac{j}{12.5}, -1 + \frac{k}{12.5} \Big)^T, \qquad i, j, k = 0...25,$$

determining the set of four-dimensional points on their graphs,

$$\Big\{ \big(\ x_i, y_j, z_k, f(x_i, y_j, z_k) \ \big)^T \mid i, j, k = 0...25 \Big\}.$$

The triangulation (tetrahedrization) for a function's graph is determined by splitting each domain cell $C_{\mathbf{i}}$ (see Definition 3.12.) specified by its eight indices in the tuple

$$\big(\ (i,j,k),\ (i+1,j,k),\ (i+1,j+1,k),\ (i,j+1,k),$$

$$(i,j,k+1),\ (i+1,j,k+1),\ (i+1,j+1,k+1),\ (i,j+1,k+1)\ \big)$$

into the six tetrahedra $T^l_{\mathbf{i}}$, $l = 1...6$, mentioned in chapter 3.2. (splitting a cuboid into six tetrahedra).

Table 4.2. summarizes the test results for the approximation of the mean and the Gaussian curvature.

Tab. 4.2. RMS errors of curvature approximation for graphs of trivariate functions.

Function	H	K
1. Linear polynomial: $.2\ (x+y+z)$.	0	0
2. Quadratic polynomial q_1: $.4\ (x^2+y^2+z^2)$.	.002950	.002597
3. Quadratic polynomial q_2: $.4\ (x^2-y^2-z^2)$.	.001115	.002216
4. Cubic polynomial: $.15\ (x^3+2x^2y-xz^2+2y^2)$.	.002545	.001207
5. Exponential function: $e^{-\frac{1}{2}(x^2+y^2+z^2)}$.	.006349	.002802
6. Trigonometric function: $.1\ \big(\cos(\pi x)+\cos(\pi y)+\cos(\pi z)\big)$.	.003269	.009065

Curvature of a trivariate function's graph is rendered by slicing the function's domain with planes and representing the magnitude of the curvature by different colors (see slicing methods, chapter 2.3.). Exact and approximated curvatures are shown for the functions 3, 4, and 6. In each figure, the exact mean and Gaussian curvatures are shown at the top, the corresponding approximated curvatures at the bottom. Figure 4.10. shows the exact and the approximated mean and Gaussian curvatures for the graph of function 3, Figure 4.11. for the graph of function 4, and Figure 4.12. for function 6.

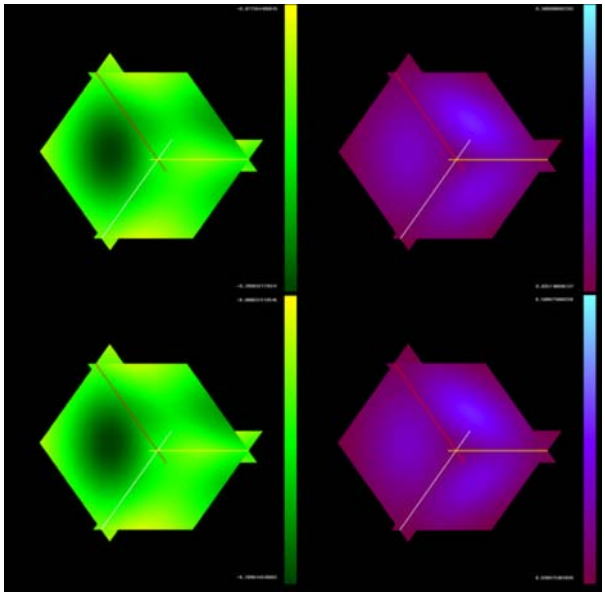

Fig. 4.10. Exact and approximated mean and Gaussian curvatures of the graph of $f(x, y, z) = .4\ (x^2 - y^2 - z^2),\ x, y, z \in [-1, 1]$.

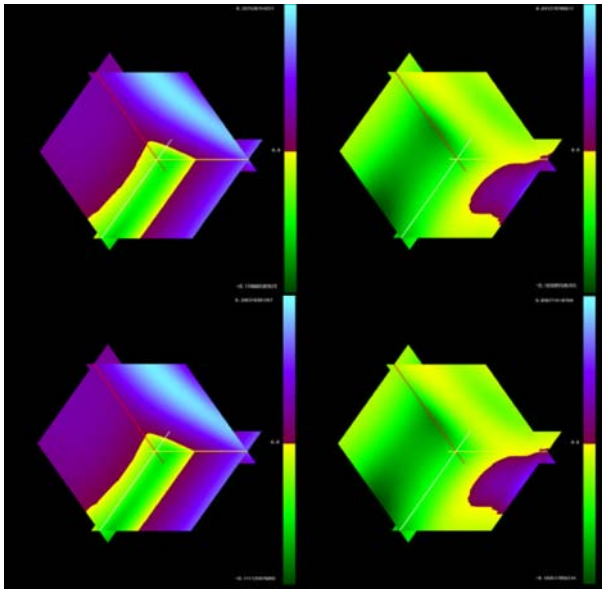

Fig. 4.11. Exact and approximated mean and Gaussian curvatures of the graph of $f(x, y, z) = .15\ (x^3 + 2x^2y - xz^2 + 2y^2),\ x, y, z \in [-1, 1]$.

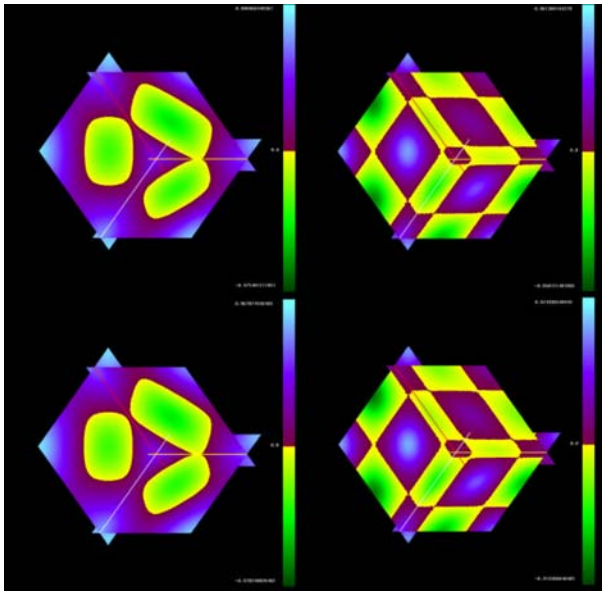

Fig. 4.12. Exact and approximated mean and Gaussian curvatures of the graph of $f(x, y, z) = .1\ \big(\cos(\pi x) + \cos(\pi y) + \cos(\pi z)\big),\ x, y, z \in [-1, 1]$.

Chapter 5

Data reduction for triangulated surfaces

5.1. Existing schemes and necessary definitions

Data reduction schemes are essential for efficient data storage. Storing and processing more data than necessary is both a waste of space and time. In the context of digitizing curves and surfaces an efficient scheme does not generate more data points than necessary to represent a particular geometric object within a prescribed tolerance. E.g., when using piecewise linear approximation storing lots of data points in "flat" regions is rather unsophisticated.

Based on this observation a data reduction algorithm is developed. Given a two-dimensional triangulation in three-dimensional space each triangle is weighted according to the principal curvatures at its vertices. A triangle indicates a surface region with low curvature, when the sum of the absolute curvatures at its vertices is low. This measure is used as a weight to determine a triangle's significance in the triangulation.

The lower a triangle's weight is the earlier it is removed. This paradigm is applied to derive an iterative algorithm removing the triangle with the lowest weight (the lowest absolute curvatures) in each step. Thus, the triangulation is adaptively modified, and the local density of triangles reflects the original surface's curvature behavior. At the end, surface regions with low curvature are represented by relatively larger triangles than highly curved regions.

The term "data-dependent triangulation" is commonly used when function

values at data points in the plane are considered for constructing a "good" triangulation of the implied piecewise linear function. This concept is discussed in [Dyn et al. '90a] and [Dyn et al. '90b]. Knot removal strategies for spline curves and tensor product surfaces (in the function setting) are described in [Arge et al. '90], [Lyche & Mørken '87], and [Lyche & Mørken '88]. Given scattered points in the plane and associated function values an iterative knot removal algorithm is discussed in [Le Méhauté & Lafranche '89] based on the resulting spline.

The data reduction technique introduced here is similar to the method in [Le Méhauté & Lafranche '89] in the sense that an iterative reduction scheme is used. However, the method deduced subsequently removes triangles instead of single data points. Furthermore, it is not restricted to a two-dimensional triangulation obtained as the graph of a bivariate function, but can be applied to more general two-dimensional triangulations in three-dimensional space, e.g., triangulations of parametric surfaces and of contours of trivariate functions.

The triangle removal algorithm allows the user to specify a percentage of the original number of triangles determining the number of triangles to be removed. Alternatively, it is possible to have the reduction process terminate automatically when a certain error tolerance is exceeded. This, however, can only be done for a triangulation obtained from a bivariate function's graph. Such a triangulation allows to compute the error introduced during each reduction step, since both the initial triangulation and the triangulation at a certain iteration step are piecewise linear functions, and their difference can easily be measured in ordinate-direction.

In principle, the method can be extended to the reduction of three-dimensional surface triangulations obtained as triangulations of three-dimensional graphs of trivariate functions in four-dimensional space. A principal curvature approximation scheme for such hypersurfaces has already been introduced in chapter 4.3. However, this extension is not investigated.

Before describing the iterative triangle removal algorithm, some necessary definitions are introduced.

Definition 5.1. Given a two-dimensional triangulation in two- or three-dimensional space, the **triangle platelet** $\mathcal{TP}_i$ associated with a triangle T_i (identified with the index triple (v_1^i, v_2^i, v_3^i) specifying its vertices) in the triangulation is the set of all triangles T_j (identified with their index triples (v_1^j, v_2^j, v_3^j)) sharing at least one of T_i's vertices as a common vertex,

$$\mathcal{TP}_i \;=\; \bigcup \left\{\; T_j \;=\; (v_1^j, v_2^j, v_3^j) \mid v_k^i = v_1^j \vee v_k^i = v_2^j \vee v_k^i = v_3^j,\; k = 1,2,3 \;\right\}. \quad (5.1.)$$

The triangle platelet $\mathcal{TP}_i$ is the set of triangles in a two-dimensional triangulation affected by the removal of the triangle T_i.

Definition 5.2. The set of triangles

$$\mathcal{CP}_i \;=\; \mathcal{TP}_i \setminus \; \{T_i\} \tag{5.2.}$$

is called the **corona** of the triangle platelet $\mathcal{TP}_i$.

Definition 5.3. The **corona** $\mathcal{CP}_i$ is **continuous** if for each pair of triangles T_{l_1}, $T_{l_m} \in \mathcal{CP}_i$ triangles $T_{l_2}, ..., T_{l_{m-1}} \in \mathcal{CP}_i$ exist such that

$$\bigwedge_{i=1}^{m-1} \big(\; T_{l_i} \text{ and } T_{l_{i+1}} \text{ are neighbors } \big); \tag{5.3.}$$

otherwise, the **corona** is **discontinuous**.

Definition 5.4. The **corona** $\mathcal{CP}_i$ is **cyclic** if it contains triangles T_{l_0}, T_{l_1}, and T_{l_2} such that

$$\bigwedge_{i=0}^{2} \big(\; T_{l_i} \text{ and } T_{l_{((i+1) \bmod 3)}} \text{ are neighbors } \big); \tag{5.4.}$$

otherwise, the **corona** is **acyclic**.

Figure 5.1. illustrates a triangle platelet $\mathcal{TP}_i$ with a continuous and an discontinuous corona and a cyclic corona.

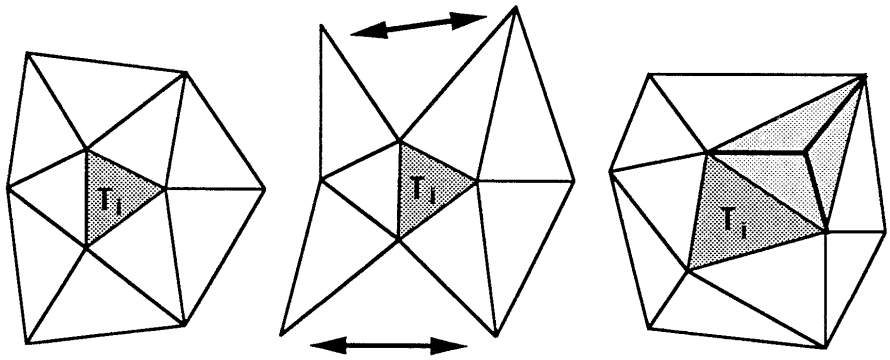

Fig. 5.1. Triangle platelet with continuous and discontinuous corona and cyclic corona.

Definition 5.5. The **corona** $\mathcal{CP}_i$ is **closed** if each triangle in $\mathcal{CP}_i$ has exactly two neighbors also elements of $\mathcal{CP}_i$; otherwise, the **corona** is called **open**.

Theorem 5.1. *Denoting the elements of a continuous, acyclic corona* $\mathcal{CP}_i$ *by* $T_{l_0}, ..., T_{l_{m_i-1}}$, *an order can be imposed on this set of triangles. If the corona* $\mathcal{CP}_i$ *is closed any triangle* $T_{l_j} \in \mathcal{CP}_i$ *among* T_i*'s neighbors can be chosen as the first triangle* T^0 *of the ordered set* $\mathcal{CP}_i^{ord}$. *If* $\mathcal{CP}_i$ *is open a triangle* $T_{l_j} \in \mathcal{CP}_i$ *not having more than one neighbor in* $\mathcal{CP}_i$ *is selected as the first triangle* T^0 *of* $\mathcal{CP}_i^{ord}$. *The set* $\mathcal{CP}_i^{ord}$ *is generated by computing the sequence of sets*

$$\mathcal{S}_0 \;=\; \{T^0\},$$

$$\mathcal{S}_k \;=\; \mathcal{S}_{k-1} \cup \{T^k\} \;=\; \{T^0, ..., T^{k-1} \mid T^j \text{ precedes } T^{j+1},\; j = 0...k-2 \;\} \cup \{T^k\},$$

$$k = 1...m_i - 1, \tag{5.5.}$$

where $T^k \in \mathcal{CP}_i$, $T^k \notin \mathcal{S}_{k-1}$, *and* T^k *is a neighbor of* T^{k-1}. *The final ordered set* $\mathcal{CP}_i^{ord}$ *equals* $\mathcal{S}_{m_i-1}$.

Proof. Trivial.

Remark 5.1. If the set of triangles to be ordered is a closed corona then the last triangle T^{m_i-1} precedes the first triangle T^0 as well.

Definition 5.6. Denoting the set of vertices in $\mathcal{TP}_i$ by $\{\mathbf{x}_{l_0}, ..., \mathbf{x}_{l_{N_i}}\}$ such that $\mathbf{a}_0 = \mathbf{x}_{l_0}$, $\mathbf{a}_1 = \mathbf{x}_{l_1}$, and $\mathbf{a}_2 = \mathbf{x}_{l_2}$ are T_i's vertices (in counterclockwise order), the set

$$\mathcal{B}_i \;=\; \{\; \mathbf{x}_{l_j} \mid j = 3...N_i \;\} \tag{5.6.}$$

is called the **boundary vertex set** of the triangle platelet $\mathcal{TP}_i$.

Theorem 5.2. *Given the ordered set of triangles $\mathcal{CP}_i^{ord}$ of a continuous, acyclic corona an order for the boundary vertex set $\mathcal{B}_i$ is implied. If the first triangle $T^0 \in \mathcal{CP}_i^{ord}$ is a neighbor of T_i the vertex of T^0 not being a vertex of T_i is chosen as the first boundary vertex $\mathbf{y}_0$ of the ordered set $\mathcal{B}_i^{ord}$. If the first triangle $T^0 \in \mathcal{CP}_i^{ord}$ is not a neighbor of T_i the vertex of T^0 neither being a vertex of T_i nor of T^1 is chosen as the first boundary vertex $\mathbf{y}_0$. The set $\mathcal{B}_i^{ord}$ is generated by computing the sequence of sets*

$$\mathcal{S}_0 \;=\; \{\mathbf{y}_0\},$$

$$\mathcal{S}_k \;=\; \mathcal{S}_{k-1} \;\cup\; \{\mathbf{y}_j\}, \qquad k = 1...m_i - 1, \tag{5.7.}$$

where $\mathbf{y}_j$ is a vertex of T^k, $\mathbf{y}_j$ is not a vertex of T_i, and $\mathbf{y}_j \notin \mathcal{S}_{k-1}$. The final ordered set $\mathcal{B}_i^{ord} = \{\mathbf{y}_0, ..., \mathbf{y}_{n_i}\}$ equals S_{m_i-1}.

Proof. Trivial.

Definition 5.7. The polygon formed by the directed line segments

$$\overline{\mathbf{y}_j\mathbf{y}_{(j+1) \bmod (n_i+1)}}, \qquad j = 0...N, \tag{5.8.}$$

where N equals $n_i - 1$ (open corona) or n_i (closed corona), is called the **platelet boundary polygon** of the triangle platelet $\mathcal{TP}_i$.

In order to ensure that the orientation of the platelet boundary polygon has the same orientation as the triangle T_i given by the line segments $\overline{\mathbf{a}_j\mathbf{a}_{(j+1) \bmod 3}}$, $j = 0, 1, 2$, the next definition is needed.

Definition 5.8. The (ordered) **boundary vertex set** $\mathcal{B}_i^{ord}$ is **ordered counterclockwise** if the platelet boundary polygon satisfies the following condition: If $\mathbf{y}_j$ and $\mathbf{y}_{(j+1) \bmod (n_i+1)}$, $j = 0...N$, are the end points of a line segment of the platelet boundary polygon and there are edges in $\mathcal{TP}_i$ connecting $\mathbf{y}_j$ with the vertex $\mathbf{a}_k$, $k \in \{0, 1, 2\}$, and $\mathbf{y}_{(j+1) \bmod (n_i+1)}$ with a different vertex $\mathbf{a}_l$, $l \in \{0, 1, 2\}$, then it is $k = 0$ and $l = 1$, or $k = 1$ and $l = 2$, or $k = 2$ and $l = 0$.

If the condition stated in Definition 5.8. is violated by the order imposed on a boundary vertex set $\mathcal{B}_i^{ord}$ the order is simply reversed. The first boundary vertex becomes the last, and the last boundary vertex becomes the first. In the following, it is assumed that both T_i's vertices and the vertices of the platelet boundary polygon are oriented counterclockwise.

Figure 5.2. illustrates different triangle platelets with platelet boundary polygons in counterclockwise order. Arrows on edges indicate the orientation.

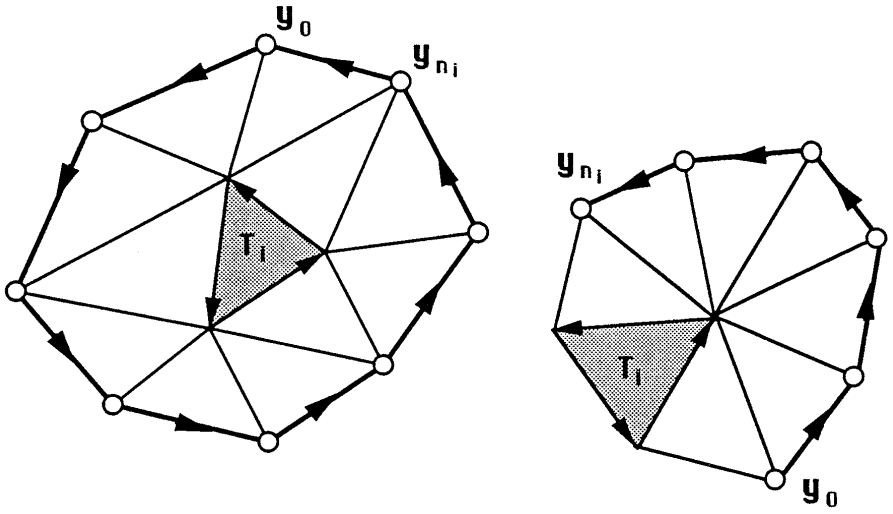

Fig. 5.2. Triangle platelet with continuous, acyclic corona and boundary vertex set.

Based on a half-plane test a criterion is introduced to decide, whether a triangle T_i in a triangulation can be removed or not. This test requires the following steps.

(i) Determine the plane equation of the plane P given by T_i.

(ii) Define an orthonormal coordinate system in P with T_i's centroid as origin and two arbitrary unit vectors in P.

(iii) Compute the distances of all points in the ordered boundary vertex set $\mathcal{B}_i^{ord}$ from P.

(iv) Project all points in $\mathcal{B}_i^{ord}$ onto P, and express the projected points with respect to the local coordinate system in P.

(v) Compute all line equations L_j in P determined by the projected directed line segments of the platelet boundary polygon.

(vi) Test, whether the centroid of T_i lies in the region obtained as the intersection of all half-planes $L_j > 0$.

Some steps are now discussed in detail. The outward unit normal vector of the plane P is given by

$$\mathbf{n} \;=\; (n^x, n^y, n^z)^T \;=\; \frac{\mathbf{d}_1 \times \mathbf{d}_2}{||\mathbf{d}_1 \times \mathbf{d}_2||}, \tag{5.9.}$$

where $\mathbf{d}_1 = \mathbf{a}_1 - \mathbf{a}_0$ and $\mathbf{d}_2 = \mathbf{a}_2 - \mathbf{a}_0$ are defined by T_i's vertices.

The plane equation for P is

$$\mathbf{n} \cdot (\mathbf{x} - \mathbf{c}) \;=\; Ax + By + Cz + D \;=\; 0, \tag{5.10.}$$

where $\mathbf{c} = (x_0, y_0, z_0)^T = \frac{1}{3}\sum_{i=0}^{2} \mathbf{a}_i$ is T_i's centroid.

The unit basis vectors for the plane P can be chosen as

$$\mathbf{b}_1 \;=\; \frac{\mathbf{d}_1}{||\mathbf{d}_1||} \qquad \text{and} \qquad \mathbf{b}_2 \;=\; \mathbf{n} \times \mathbf{b}_1. \tag{5.11.}$$

As in chapter 4.2., the signed distances d_j, $j = 0...n_i$, of the platelet boundary points $\mathbf{y}_j = (x_j, y_j, z_j)^T$ are

$$d_j \;=\; Ax_j + By_j + Cz_j + D. \tag{5.12.}$$

Projecting the platelet boundary points onto P yields the points $\mathbf{y}_j^P$, where

$$\mathbf{y}_j^P \;=\; \mathbf{y}_j - d_j\ \mathbf{n}. \tag{5.13.}$$

Expressing the points $\mathbf{y}_j^P$ in P with respect to the two-dimensional coordinate system given by its origin $\mathbf{c}$ and the two basis vectors $\mathbf{b}_1$ and $\mathbf{b}_2$, one obtains

$$\mathbf{d}_j \;=\; (\mathbf{d}_j \cdot \mathbf{b}_1)\;\mathbf{b}_1 + (\mathbf{d}_j \cdot \mathbf{b}_2)\;\mathbf{b}_2, \tag{5.14.}$$

where $\mathbf{d}_j = \mathbf{y}_j^P - \mathbf{c}$, $j = 0...n_i$. Therefore, the local coordinates $(u_j, v_j)^T$ of a point $\mathbf{y}_j^P$ with respect to the planar coordinate system are given by

$$\big(u_j, v_j\big)^T \;=\; \big(\mathbf{d}_j \cdot \mathbf{b}_1, \mathbf{d}_j \cdot \mathbf{b}_2\big)^T. \tag{5.15.}$$

Projected onto P, the points $\mathbf{y}_j^P$ form an oriented polygon as well. The line equations of the single segments are expressed using the local planar coordinate system. The implicit line equation for the line $L_j(u, v)$ is given by

$$L_j\;(u, v) \;=\; -\;\triangle v_j\;(u - u_j)\;+\;\triangle u_j\;(v - v_j)\;=\;0, \tag{5.16.}$$

where $\triangle u_j = u_{(j+1)\bmod(n_i+1)} - u_j$ and $\triangle v_j = v_{(j+1)\bmod(n_i+1)} - v_j$, $j = 0...N$.

Now, the criterion is given to decide, whether the triangle T_i can be removed. If the centroid $\mathbf{c}$ (with local coordinates $(0, 0)^T$) is on the "left," "positive" side of all lines L_j the triangle can be removed.

Definition 5.9. The solution set of the $N + 1$ linear inequalities

$$L_j\;(u, v) \;=\; -\;\triangle v_j\;(u - u_j)\;+\;\triangle u_j\;(v - v_j)\;>\;0, \tag{5.17.}$$

$j = 0...N$, is called the **feasible region** of the triangle platelet $\mathcal{TP}_i$ in the plane P.

Theorem 5.3. *The centroid* $\mathbf{c}$ *of a triangle* T_i *is in the feasible region defined by the inequalities in (5.17.) if the inequalities*

$$u_j\;\triangle v_j\;-\;v_j\;\triangle u_j\;>\;0, \tag{5.18.}$$

$j = 0...N$, *hold for all* j.

Proof. Assuming that the intersection of all half-planes defined by (5.17.) is not empty and inserting the local coordinates $(0, 0)^T$ of the centroid $\mathbf{c}$ into (5.17.) proves the theorem.

q.e.d.

Remark 5.2. A triangle T_i can only be removed if it is surrounded by a continuous, acyclic corona, and its centroid passes the planar half-plane test.

In Figure 5.3., the half-plane test applied to the centroid of a triangle T_i passing the test is shown.

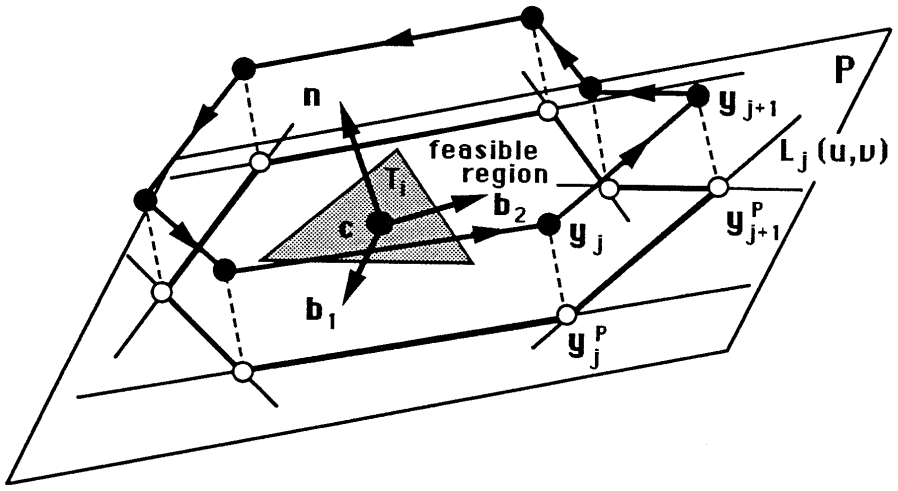

Fig. 5.3. Boundary vertex set and its projection onto plane P; triangle T_i passing the half-plane test.

5.2. Triangle reduction for triangulated two-dimensional surfaces

In order to determine the significance (weight) of a triangle in a two-dimensional triangulation, the principal curvatures at its vertices and its interior angles are considered. In this context, neither mean nor Gaussian curvature serve well as measures for a triangle's significance. The mean curvature at the point $(0,0,0)^T$ on the hyperboloid $\left(x, y, x^2 - y^2\right)^T$, $x, y \in \mathbb{R}$, is zero, and the Gaussian curvature at points in a plane and on a cylinder are both zero. Therefore, absolute curvature is used as an appropriate curvature measure.

Definition 5.10. The sum A of the absolute values of the principal curvatures κ_1 and κ_2 at a point $\mathbf{x}_0$ on the regular parametric surface $\mathbf{x}(\mathbf{u})$ is called the **absolute curvature**,

$$A \;=\; |\kappa_1| \;+\; |\kappa_2|. \tag{5.19.}$$

Since the overall goal is to establish an order in increasing significance on the finite set of triangles constituting a two-dimensional triangulation in three-dimensional space, each triangle is weighted by the three absolute curvatures at its vertices. The triangle with minimal absolute curvature is least significant, while the triangle with maximal absolute curvature is most significant. Later, the triangles are iteratively removed from the triangulation according to this order.

Lemma 5.1. *Denoting the interior angles of a triangle by* α_1, α_2, *and* α_3, *the range*

of the function

$$f\ (\alpha_1, \alpha_2, \alpha_3)\ =\ \sum_{j=1}^{3} \cos \alpha_j \tag{5.20.}$$

is the interval $[1, \frac{3}{2}]$.

Proof. Since $\sum_{j=1}^{3} \alpha_j = \pi$, it is sufficient to analyze the bivariate function

$$g\ (\alpha_1, \alpha_2)\ =\ \cos \alpha_1 + \cos \alpha_2 + \cos\big(\pi - (\alpha_1 + \alpha_2)\big)$$

on the domain $D = \big\{\ (\alpha_1, \alpha_2)\ |\ \alpha_1, \alpha_2 \geq 0,\ \alpha_1 + \alpha_2 \leq \pi\ \big\}$.

One easily proves that g equals one on D's boundary:

$$g\ (0, \alpha_2)\ =\ 1 + \cos \alpha_2 + \cos\ (\pi - \alpha_2)\ =\ 1,$$

$$g\ (\alpha_1, 0)\ =\ \cos \alpha_1 + 1 + \cos\ (\pi - \alpha_1)\ =\ 1, \quad \text{and}$$

$$g\ (\alpha_1, \pi - \alpha_1)\ =\ \cos \alpha_1 + \cos\ (\pi - \alpha_1) + 1\ =\ 1.$$

A critical point must satisfy the equations

$$\frac{\partial g}{\partial \alpha_1}\ (\alpha_1, \alpha_2)\ =\ -\sin \alpha_1 + \sin\big(\pi - (\alpha_1 + \alpha_2)\big)\ =\ 0 \quad \text{and}$$

$$\frac{\partial g}{\partial \alpha_2}\ (\alpha_1, \alpha_2)\ =\ -\sin \alpha_2 + \sin\big(\pi - (\alpha_1 + \alpha_2)\big)\ =\ 0.$$

Therefore, $\sin \alpha_1 = \sin \alpha_2$ is a necessary condition which holds for $\alpha_1 = \alpha_2$ and $\alpha_2 = \pi - \alpha_1$.

The first case, $\alpha_1 = \alpha_2$, defines the univariate function

$$h\ (\alpha_1)\ =\ 2\ \cos \alpha_1 + \cos\ (\pi - 2\alpha_1)$$

having critical points at $\alpha_1 = 0$ and $\alpha_1 = \frac{\pi}{3}$, since $h'(0) = h'(\frac{\pi}{3}) = 0$. Considering only $\alpha_1 = \frac{\pi}{3}$ results in the function value $f(\frac{\pi}{3}, \frac{\pi}{3}, \frac{\pi}{3}) = \frac{3}{2}$.

The second case, $\alpha_2 = \pi - \alpha_1$, defines part of D's boundary, where f equals one.

q.e.d.

Definition 5.11. The **angle weight σ_i of a triangle** T_i is given by

$$\sigma_i \;=\; \sigma\,(T_i) \;=\; 2\,\Big(\;\big(\;\sum_{j=1}^{3}\cos\alpha_j\;\big) - 1\;\Big) \quad \in\; [0,1], \tag{5.21.}$$

where α_j, $j = 1, 2, 3$, are T_i's interior angles.

Remark 5.3. The weight function in equation (5.21.) assigns maximum weight to an equilateral triangle and small weights to "long," "skinny" triangles.

Definition 5.12. The **curvature weight ρ_i of a triangle** T_i is given by the sum of the absolute curvatures at its vertices,

$$\rho_i \;=\; \rho\,(T_i) \;=\; \sum_{j=1}^{3} A_j, \tag{5.22.}$$

where A_j, $j = 1, 2, 3$, are the absolute curvatures at T_i's vertices.

Definition 5.13. The **weight ω_i of a triangle** T_i is given by

$$\omega_i \;=\; \omega\,(T_i) \;=\; \sigma_i\;\rho_i. \tag{5.23.}$$

The different steps concerning the removal of a single triangle T_i are discussed next. Assuming that the triangle platelet $\mathcal{TP}_i$ satisfies all the conditions stated in chapter 5.1., the triangle T_i is removed from the triangulation by replacing its

vertices by one new point **p** whose construction must be described. This new point is connected to each point in $\mathcal{TP}_i$'s boundary vertex set, thus determining the new edges in the triangulation.

It is worth mentioning that this method of replacing a triangle does not affect the genus of the triangulation (precisely, the genus of the triangulated surface).

Definition 5.14. Given a two-dimensional triangulation $\mathcal{T}$, where each triangle has exactly three neighbors, the value

$$\chi \;=\; t - e + v, \tag{5.24.}$$

where t is the number of triangles, e the number of edges, and v the number of vertices in $\mathcal{T}$, is called the **Euler-Poincaré** characteristic of the implied C^0, piecewise linear surface. The **topological genus** is the value

$$1 - \frac{\chi}{2}. \tag{5.25.}$$

Remark 5.4. Considering a triangulation including triangles not having exactly three neighbors, equation (5.24.) must be modified. In this case, χ is defined as $t - e + v + 1$ ("open triangulation").

Theorem 5.4. *Replacing a triangle T_i whose corona $\mathcal{CP}_i$ is continuous and acyclic by a point* **p**, *and constructing new edges by connecting the new point with each point in the boundary vertex set $\mathcal{B}_i^{ord}$, preserves the Euler-Poincaré characteristic.*

Proof. Replacing T_i by a point obviously reduces the number of vertices by two. Let k denote the number of edges (locally) being removed from the triangulation.

Then, the number of triangles is reduced by $(k-2)$. Considering these numbers and inserting them into equation (5.24.) yields

$$\chi \;=\; \big(t-(k-2)\big)-\big(e-k\big)+\big(v-2\big) \;=\; t-e+v,$$

proving that the Euler-Poincaré characteristic remains the same.

q.e.d.

In principle, there are two possibilities for the construction of the new point $\mathbf{p}$ replacing the triangle T_i. One possibility is to construct a bivariate function $f(u,v)$ using an appropriate coordinate system for the points determining T_i's triangle platelet and to evaluate f at $(0,0)^T$. The other possibility is to compute an implicit function $f(x,y,z)=0$, considering the same set of data points and to generate the new point by intersecting a line with the implicitly defined surface.

The first possibility is described next. The construction follows the same principle as the half-plane test (see chapter 5.1.), and the nomenclature from there is used. Depending on the corona $\mathcal{CP}_i^{ord}$, different choices for the origin $\mathbf{c}$ in the plane P are made.

- If the corona $\mathcal{CP}_i^{ord}$ is closed then the centroid of T_i is chosen.
- If T_i has three neighbors, but its corona is open, then the common vertex of T_i and the first and last triangle in $\mathcal{CP}_i^{ord}$ is chosen.
- If T_i has two neighbors, and the first and last triangle in $\mathcal{CP}_i^{ord}$ are both (are both not) neighbors of T_i then the mid-point of T_i's edge not shared by another triangle is chosen.

- If T_i has two neighbors, and the first (the last) triangle in $\mathcal{CP}_i^{ord}$ is a neighbor of T_i, and the last (the first) triangle in $\mathcal{CP}_i^{ord}$ is not a neighbor of T_i, then the vertex only shared by T_i and the first (the last) triangle in $\mathcal{CP}_i^{ord}$ is chosen.
- If T_i has a vertex not shared by another triangle then that vertex is chosen.

The different choices for the origin **c** are shown in Figure 5.4.

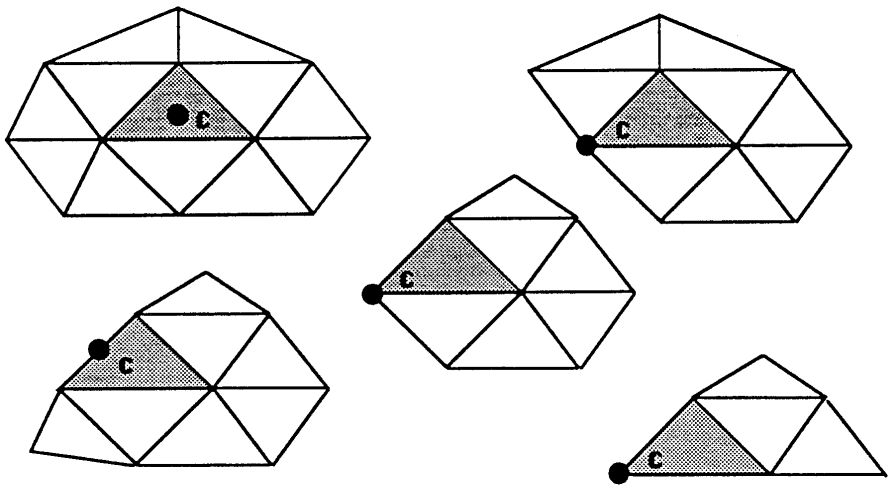

Fig. 5.4. Different choices for origin depending on triangle platelet.

Denoting the set of vertices in T_i's triangle platelet by $\{\mathbf{x}_1, ..., \mathbf{x}_{n_i}\}$, their associated two-dimensional coordinates in the plane P by $(u_j, v_j)^T$, and their distances from P by d_j, $j = 1...n_i$, again, a polynomial of degree two is constructed consider-

ing the constraints

$$f\ (u_j, v_j)\ =\ \sum_{\substack{i+k\leq 2\\ i,k\geq 0}} c_{i,k}\ \ {u_j}^i\ {v_j}^k\ =\ d_j, \qquad j = 1...n_i. \tag{5.26.}$$

In matrix representation these constraints are

$$\begin{pmatrix} {u_1}^2 & u_1v_1 & u_1 & {v_1}^2 & v_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ {u_{n_i}}^2 & u_{n_i}v_{n_i} & u_{n_i} & {v_{n_i}}^2 & v_{n_i} & 1 \end{pmatrix} \begin{pmatrix} c_{2,0} \\ c_{1,1} \\ c_{1,0} \\ c_{0,2} \\ c_{0,1} \\ c_{0,0} \end{pmatrix} \ =\ U\ \mathbf{c}\ =\ \mathbf{d}\ =\ \begin{pmatrix} d_1 \\ \vdots \\ d_{n_i} \end{pmatrix}. \tag{5.27.}$$

Solving the normal equations

$$U^T\ U\ \mathbf{c}\ =\ U^T\ \mathbf{d} \tag{5.28.}$$

finally determines a local approximation in a function setting, provided that the determinant of U^TU does not vanish.

The new point $\mathbf{p}$ by which the triangle T_i is replaced is the point

$$\mathbf{p}\ =\ \mathbf{c} + f(0,0)\ \mathbf{n}, \tag{5.29.}$$

where $\mathbf{n}$ is T_i's outward unit normal vector (ordinate direction of f).

The second possibility to determine the new point $\mathbf{p}$ is the construction of a quadric $f(x, y, z) = 0$ obtained by considering the constraints

$$f\ (x_j, y_j, z_j)\ =\ \sum_{\substack{i+k+l\leq 2\\ i,k,l\geq 0}} c_{i,k,l}\ \ {x_j}^i\ {y_j}^k\ {z_j}^l\ =\ 0, \qquad j = 1...n_i,$$

and the additional linear constraint

$$f\ (1,1,1)\ =\ \sum_{\substack{i+k+l\leq 2\\ i,k,l\geq 0}} c_{i,k,l}\ =\ 1, \tag{5.30.}$$

where $(x_j, y_j, z_j)^T \in \{\mathbf{x}_1, ..., \mathbf{x}_{n_i}\}$ and $(1,1,1)^T \notin \{\mathbf{x}_1, ..., \mathbf{x}_{n_i}\}$, and $\{\mathbf{x}_1, ..., \mathbf{x}_{n_i}\}$ is the set of vertices in the triangle platelet $\mathcal{TP}_i$.

These conditions can be rewritten as

$$\begin{pmatrix} {x_1}^2 & x_1 y_1 & \dots & z_1 & 1 \\ {x_2}^2 & x_2 y_2 & \dots & z_2 & 1 \\ \vdots & \vdots & & \vdots & \vdots \\ {x_{n_i}}^2 & x_{n_i} y_{n_i} & \dots & z_{n_i} & 1 \\ 1 & 1 & \dots & 1 & 1 \end{pmatrix} \begin{pmatrix} c_{2,0,0} \\ c_{1,1,0} \\ \vdots \\ c_{0,0,1} \\ c_{0,0,0} \end{pmatrix} = X\ \mathbf{c} = \mathbf{y} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \qquad (5.31.)$$

Solving the normal equations

$$X^T\ X\ \mathbf{c} = X^T\ \mathbf{y} \qquad (5.32.)$$

finally determines a quadric surface locally approximating the vertices in the triangle platelet $\mathcal{TP}_i$.

Remark 5.5. The additional constraint $f(1,1,1) = 1$ is added, since the equation $f(x,y,z) = 0$ can be multiplied by any scalar still describing the same surface. Also, the condition $f(1,1,1) = 1$ defines an orientation of the quadric surface, its "outside" and "inside." However, this does not affect the absolute curvature at any point on the quadric.

Remark 5.6. If a triangle platelet does not provide enough vertices to uniquely determine the coefficients for a quadric (at least nine vertices) an appropriate subset of implicit surfaces is chosen, e.g.,

$$f\ (x,y,z) = \sum_{\substack{i+k+l \le 2 \\ 0 \le i,k,l \le 1}} c_{i,k,l}\ \ x^i\ y^k\ z^l,$$

requiring less vertices.

The new point $\mathbf{p}$ is computed by intersecting the quadric surface with the line $\mathbf{x}(t) = \mathbf{c} + t\mathbf{n}$, $t \in I\!R$, where $\mathbf{c}$ is constructed as in the bivariate function setting, and $\mathbf{n}$ is T_i's outward unit normal vector. Inserting the linear expressions for the single components $x(t)$, $y(t)$, and $z(t)$ into the equation

$$f\ \big(x(t), y(t), z(t)\big)\ =\ \sum_{\substack{i+k+l\leq 2\\ i,k,l\geq 0}} c_{i,k,l}\ \ \big(x(t)\big)^i\ \big(y(t)\big)^k\ \big(z(t)\big)^l\ =\ 0$$

yields the quadratic equation

$$\begin{aligned} t^2\ \Big(\ & c_{2,0,0}\ (n^x)^2 + c_{1,1,0}\ n^x n^y + c_{1,0,1}\ n^x n^z + c_{0,2,0}\ (n^y)^2 + c_{0,1,1}\ n^y n^z + c_{0,0,2}\ (n^z)^2\ \Big) \\ & + t\ \Big(\ 2\ (\ c_{2,0,0}\ c^x n^x + c_{0,2,0}\ c^y n^y + c_{0,0,2}\ c^z n^z\ ) \\ & + c_{1,1,0}\ (c^x n^y + c^y n^x) + c_{1,0,1}\ (c^x n^z + c^z n^x) + c_{0,1,1}\ (c^y n^z + c^z n^y) \\ & + c_{1,0,0}\ n^x + c_{0,1,0}\ n^y + c_{0,0,1}\ n^z\ \Big) \\ & + \Big(\ c_{2,0,0}\ (c^x)^2 + c_{1,1,0}\ c^x c^y + c_{1,0,1}\ c^x c^z + c_{1,0,0}\ c^x + c_{0,2,0}\ (c^y)^2 \\ & + c_{0,1,1}\ c^y c^z + c_{0,1,0}\ c^y + c_{0,0,2}\ (c^z)^2 + c_{0,0,1}\ c^z + c_{0,0,0}\ \Big)\ \ =\ \ 0, \end{aligned} \qquad (5.33.)$$

where $\mathbf{c} = (c^x, c^y, c^z)^T$ and $\mathbf{n} = (n^x, n^y, n^z)^T$.

Denoting the two (real) solutions of this equation by t_1 and t_2, the point in $\{\ \mathbf{c} + t_i\mathbf{n}\ \mid\ i = 1, 2\ \}$ having minimal distance to the plane P spanned by T_i is selected as the new point $\mathbf{p}$. Should the discriminant of the quadratic equation become negative, $\mathbf{c}$ is chosen as the point $\mathbf{p}$.

Having computed $\mathbf{p}$ by either the bivariate setting or by the implicit, trivariate approach, a first local re-triangulation of the boundary vertex set and $\mathbf{p}$ is obtained by connecting each vertex in $\mathcal{B}_i^{ord}$ with $\mathbf{p}$.

Figure 5.5. illustrates the removal of a triangle T_i with different triangle platelets and the re-triangulation of the remaining platelet boundary vertex set and the new point **p**.

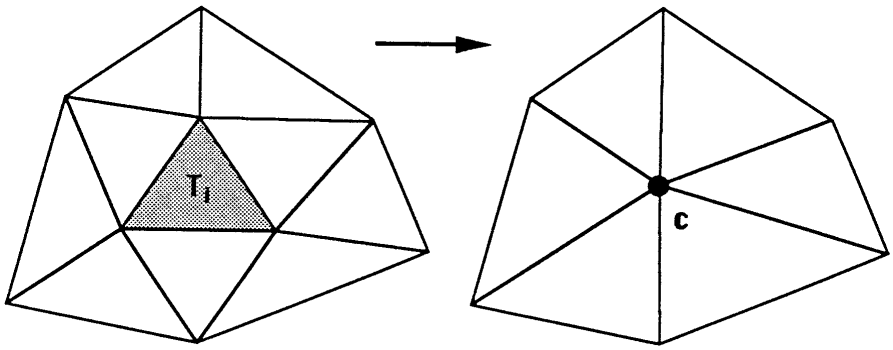

Fig. 5.5. Removal of triangle T_i and re-triangulation of boundary vertex set and new point.

Regardless of the method used for determining the new vertex **p**, the absolute curvature must be computed for it, since the newly constructed triangles need to be weighted and appropriately inserted into the overall order of all triangles.

Assuming the new point **p** has been constructed by the bivariate function approach, the following theorem holds.

Theorem 5.5. *The principal curvatures* κ_1 *and* κ_2 *of the graph* $\big(\, u, v, f(u,v) \,\big)^T \subset \mathbb{R}^3$, $u, v \in \mathbb{R}$, *of the bivariate polynomial*

$$f\ (u,v)\ =\ \sum_{\substack{i+k\leq 2 \\ i,k\geq 0}} c_{i,k}\ \ {u_j}^i\ {v_j}^k \tag{5.34.}$$

at the point $\left(\, 0, 0, f(0,0) \, \right)^T$ *are given by the two real roots of the quadratic equation*

$$\det \begin{pmatrix} 2\ c_{2,0}\ (1+{c_{0,1}}^2) - c_{1,1}c_{1,0}c_{0,1} - \kappa & -2\ c_{2,0}c_{1,0}c_{0,1} + c_{1,1}\ (1+{c_{1,0}}^2) \\ c_{1,1}\ (1+{c_{0,1}}^2) - 2\ c_{0,2}c_{1,0}c_{0,1} & -c_{1,1}c_{1,0}c_{0,1} + 2\ c_{0,2}\ (1+{c_{1,0}}^2) - \kappa \end{pmatrix}$$

$$= \ 0. \qquad (5.35.)$$

Proof. According to Definition 4.7. and equation (4.15.) the principal curvatures of f's graph are the eigenvalues of the matrix

$$-A \ = \ \frac{1}{l_1} \begin{pmatrix} f_{uu} & f_{uv} \\ f_{uv} & f_{vv} \end{pmatrix} \begin{pmatrix} 1+{f_u}^2 & f_u f_v \\ f_u f_v & 1+{f_v}^2 \end{pmatrix}^{-1},$$

where $l_1 = \sqrt{1+{f_u}^2+{f_v}^2}$. Evaluating $-A$ for $u = v = 0$, one obtains the matrix

$$\frac{1}{l_1} \begin{pmatrix} 2\ c_{2,0} & c_{1,1} \\ c_{1,1} & 2\ c_{0,2} \end{pmatrix} \begin{pmatrix} 1+{c_{1,0}}^2 & c_{1,0}c_{0,1} \\ c_{1,0}c_{0,1} & 1+{c_{0,1}}^2 \end{pmatrix}^{-1},$$

where $l_1 = \sqrt{1+{c_{1,0}}^2+{c_{0,1}}^2}$, having the characteristic equation (5.35.).

q.e.d.

The two roots of equation (5.35.), κ_1 and κ_2, determine the absolute curvature at the point $\mathbf{p}$ and therefore the curvature weights for all triangles sharing $\mathbf{p}$ as a common vertex.

Choosing the other possibility for computing $\mathbf{p}$, intersecting a quadric with a line, a formula is needed for calculating the principal curvatures for an arbitrary point on a quadric. It is well known in differential geometry how to compute the principal curvatures on an implicitly defined surface $f(x,y,z) = 0$.

Theorem 5.6. *Given the implicitly defined surface*

$$\mathcal{S} \ = \ \left\{ \ \mathbf{x} \in I\!R^3 \mid f(\mathbf{x}) = 0 \ \right\},$$

where f is some polynomial with a non-vanishing gradient $\bigtriangledown f$ for all points in $\mathcal{S}$, and defining the surface outward normal vector at the surface point $\mathbf{x}_0 = (x_0, y_0, z_0)^T \in \mathcal{S}$ as

$$\mathbf{n} \;=\; \frac{1}{2}\;\left(\;\bigtriangledown f\;(x_0, y_0, z_0)\;\right)^T$$

the tangent space at $\mathbf{x}_0$ is spanned by the two vectors $\mathbf{b}_1$ and $\mathbf{b}_2$, where $\mathbf{b}_1$ is any unit vector perpendicular to $\mathbf{n}$ and $\mathbf{b}_2$ is the normalized cross product of $\mathbf{n}$ and $\mathbf{b}_1$,

$$\mathbf{b}_1 \cdot \mathbf{n} \;=\; 0, \quad \sqrt{\mathbf{b}_1 \cdot \mathbf{b}_1} \;=\; 1, \qquad \textit{and} \qquad \mathbf{b}_2 \;=\; \mathbf{n} \times \mathbf{b}_1 \frac{}{||\mathbf{n} \times \mathbf{b}_1||}. \qquad (5.36.)$$

Introducing the vector valued differential operator $\mathbf{D_d}$ as

$$\mathbf{D_d}\; f \;\Big|_{\mathbf{x}_0} \;=\; \frac{1}{2}\begin{pmatrix} f_{xx}d^x + f_{xy}d^y + f_{xz}d^z \\ f_{xy}d^x + f_{yy}d^y + f_{yz}d^z \\ f_{xz}d^x + f_{yz}d^y + f_{zz}d^z \end{pmatrix} \;\Big|_{\mathbf{x}_0}, \qquad (5.37.)$$

where $\mathbf{d} = (d^x, d^y, d^z)^T$ is a directional vector and $f_{xx}, ..., f_{zz}$ denote the second order partial derivatives of f, the mean and Gaussian curvature at $\mathbf{x}_0$ are

$$H \;=\; -\;\frac{\mathbf{n}\;\cdot\;\left(\mathbf{D}_{\mathbf{b}_1} f \times \mathbf{b}_2 \;+\; \mathbf{b}_1 \times \mathbf{D}_{\mathbf{b}_2} f\right)}{2\;||\;\mathbf{n}\;||^{\;3}} \;\Big|_{\mathbf{x}_0} \qquad \textit{and}$$

$$K \;=\; \frac{\mathbf{n}\;\cdot\;\left(\mathbf{D}_{\mathbf{b}_1} f \times \mathbf{D}_{\mathbf{b}_2} f\right)}{||\;\mathbf{n}\;||^{\;4}} \;\Big|_{\mathbf{x}_0}, \qquad (5.38.)$$

where $||\mathbf{n}|| = \sqrt{\mathbf{n} \cdot \mathbf{n}}$. The principal curvatures, κ_1 and κ_2, are related to the mean and Gaussian curvature by

$$\kappa_{1/2} \;=\; H \;\pm\; \sqrt{H^2 - K}. \qquad (5.39.)$$

Proof. See [O'Neill '69], chapter 5.

These formulae can easily be applied to a quadric, thus determining the absolute curvature at the new point $\mathbf{p}$.

As already mentioned above, a natural way to triangulate the platelet boundary vertex set $\mathcal{B}_i^{ord}$ and the additional, new point $\mathbf{p}$ is to construct edges from $\mathbf{p}$ to each vertex in $\mathcal{B}_i^{ord}$, thus defining the (local) re-triangulation $\mathcal{N}_i$,

$$\mathcal{N}_i \;=\; \bigcup \{\; T_j \;=\; (l_0, l_k, l_{(k+1)\bmod(N_i+1)}) \mid k = 3...m \;\}, \tag{5.40.}$$

where m equals either $N_i - 1$ (open corona) or N_i (closed corona), and $\{\mathbf{x}_{l_0}, ..., \mathbf{x}_{l_{N_i}}\}$ is the set of vertices in $\mathcal{TP}_i$ (see Definition 5.6.). This means that the new point $\mathbf{p}$ "inherits" the index of the first vertex, l_0, of the removed triangle, T_i, and vertices with indices l_1 and l_2 no longer exist.

In order to obtain a (local) re-triangulation consisting of triangles with high angle weights, an iterative, Lawson-like algorithm is applied to the set of newly constructed triangles in the set $\mathcal{N}_i$, therefore iteratively modifying the triangulation $\mathcal{N}_i$ (see [Lawson '77]). The idea is to swap diagonals of quadrilaterals, edges shared by two neighbors in $\mathcal{N}_i$. Nevertheless, diagonals are swapped only if the region obtained by projecting a quadrilateral onto the plane P determined by the removed triangle T_i is convex.

Definition 5.15. The **quadrilateral** formed by the line segments $\overline{\mathbf{ab}}$, $\overline{\mathbf{bc}}$, $\overline{\mathbf{cd}}$, $\overline{\mathbf{da}}$, $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d} \in I\!R^3$, **has a convex projection with respect to a plane** P if the quadrilateral in P, formed by the line segments $\overline{\mathbf{a}^P\mathbf{b}^P}$, $\overline{\mathbf{b}^P\mathbf{c}^P}$, $\overline{\mathbf{c}^P\mathbf{d}^P}$, $\overline{\mathbf{d}^P\mathbf{a}^P}$, where $\mathbf{a}^P$, $\mathbf{b}^P$, $\mathbf{c}^P$, and $\mathbf{d}^P$ are the orthogonal projections of $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{d}$ onto P,

describes the polygonal boundary of a convex region in P.

Quadrilaterals formed by neighbor triangles in $\mathcal{N}_i$ satisfying this condition are swapped as long as this results in an increase of the minimum of the angle weights in the set of new triangles. This strategy finally terminates, since there is only a limited number of possible triangulations and one of them maximizes the minimum angle weight in $\mathcal{N}_i$.

Figure 5.6. shows the effect of swapping diagonals in a local re-triangulation, demonstrating the improvement of angle weights.

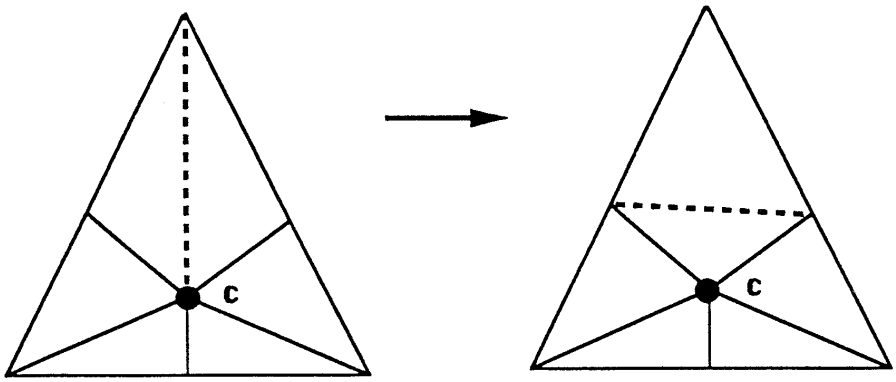

Fig. 5.6. Increasing angle weights of triangles in local re-triangulation (original and improved re-triangulation).

Having computed the weights of all triangles in the set $\mathcal{N}_i$ they are inserted into the overall order of all triangles. Simplified, the triangle reduction algorithm can be summarized as follows.

Algorithm 5.1. *Triangle reduction by iterative triangle removal*

Input: table $\mathcal{T}$ of N_0 triangles (including neighborhood information), table $\mathcal{V}$ of vertices (including principal curvatures), and a percentage $p \in [0, 100]$.
Output: reduced table $\hat{\mathcal{T}}$ of triangles and reduced table $\hat{\mathcal{V}}$ of vertices.

compute weights for each triangle in $\mathcal{T}$;
while number of triangles is greater than $\frac{p}{100}\ N_0$
(
among all triangles having a continuous, acyclic corona and passing the half-plane test determine the triangle T_i with minimal weight ω_{min};
remove triangle T_i from triangulation (using either a bivariate or a trivariate, implicit least squares approximation);
compute a first (local) re-triangulation;
compute the curvature weights for all new triangles;
improve the (local) re-triangulation by maximizing the minimum angle weight;
compute weights for new triangles;
)

Remark 5.7. If two triangles T_i and T_j exist both having minimum weight ω_{min} any of the two can be removed first. Removing either one of them first does not affect the final result as long as the triangle platelets $\mathcal{TP}_i$ and $\mathcal{TP}_j$ have an empty intersection.

Remark 5.8. Considering triangles not surrounded by a closed corona as boundary triangles of a triangulation, it is possible to force the algorithm not to remove such triangles. This, however, leads to reduced triangulations keeping a high density of vertices on the boundary. This problem, of course, does not arise in the case of reducing triangulations of closed surfaces.

Remark 5.9. In each iteration step triangles obtained by the local re-triangulation procedure are marked as "new" triangles which can not be removed at once in the

next iteration step. Only if no triangle among the "old" ones can be removed, all new triangles can be considered for removal.

Remark 5.10. The termination criterion in the above algorithm (using a certain percentage of the original number of triangles) can be modified in the case of a triangulation obtained from the graph of a bivariate function. Then, the RMS error can be computed interpreting original and an intermediate triangulation as piecewise linear functions. As soon as the RMS error exceeds a certain tolerance ϵ, the algorithm stops.

Remark 5.11. It is possible that the reduction algorithm can not find any triangle having a continuous, acyclic corona and passing the half-plane test.

In the following examples, a triangle is always replaced by a point using the bivariate function approach for computing a new point $\mathbf{p}$. Each of the next three figures shows an original (upper-left) and three reduced triangulations for a reduction in the number of triangles by 50% (upper-right), 80% (lower-left), and 90% (lower-right). The initial triangulations for Figures 5.7., 5.8., and 5.9. are obtained by evaluating particular bivariate functions on $[-1,1] \times [-1,1]$ using a domain grid with equidistant spacing determining points on their graphs.

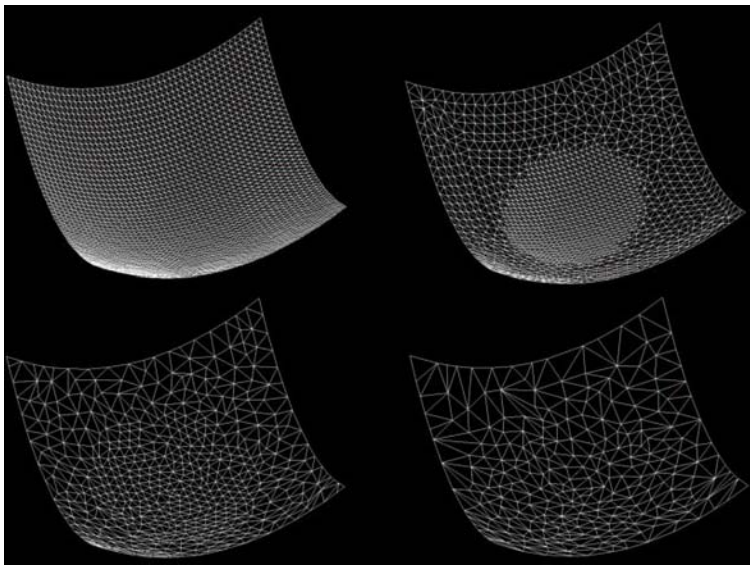

Fig. 5.7. Triangle reduction of 50%, 80%, and 90%
for the graph of $f(x, y) = .4\ (x^2 + y^2),\ x, y \in [-1, 1]$.

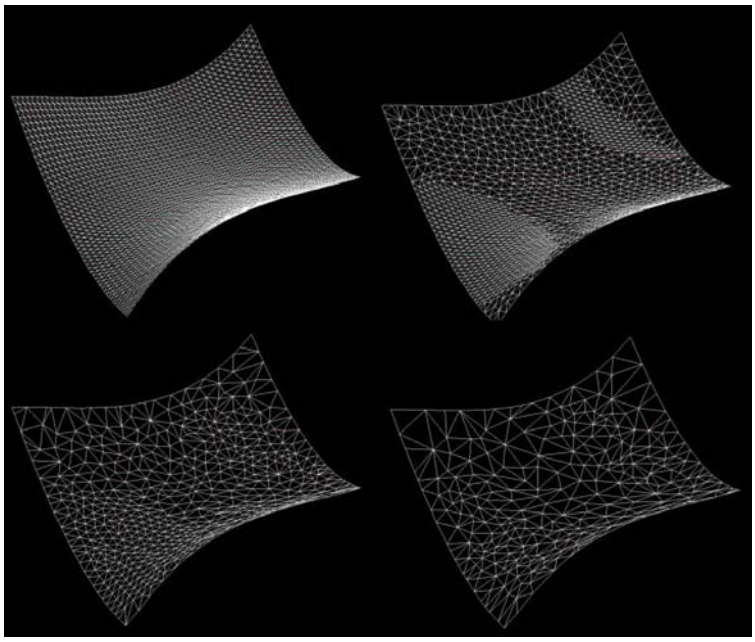

Fig. 5.8. Triangle reduction of 50%, 80%, and 90%
for the graph of $f(x, y) = .15\ (x^3 + 2x^2y - xy + 2y^2),\ x, y \in [-1, 1]$.

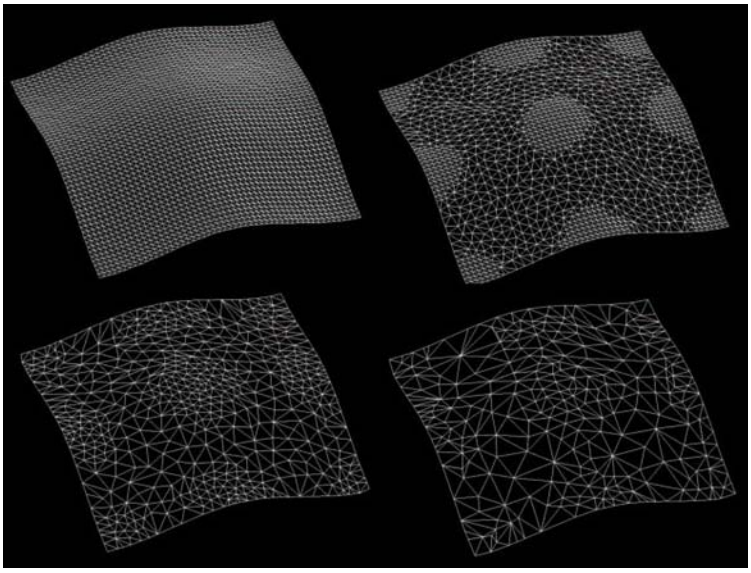

Fig. 5.9. Triangle reduction of 50%, 80%, and 90%
for the graph of $f(x,y) = .1\ \big(\cos(\pi x) + \cos(\pi y)\big),\ x, y \in [-1, 1]$.

Figure 5.10. shows the reduction algorithm applied to a torus and Figure 5.11. shows the original triangular approximation to a human skull (left, about 60,000 triangles obtained by computing a triangular approximation of a particular contour level of a CAT scan data set) and the result after a reduction by 90% (right). All triangles are flat-shaded.

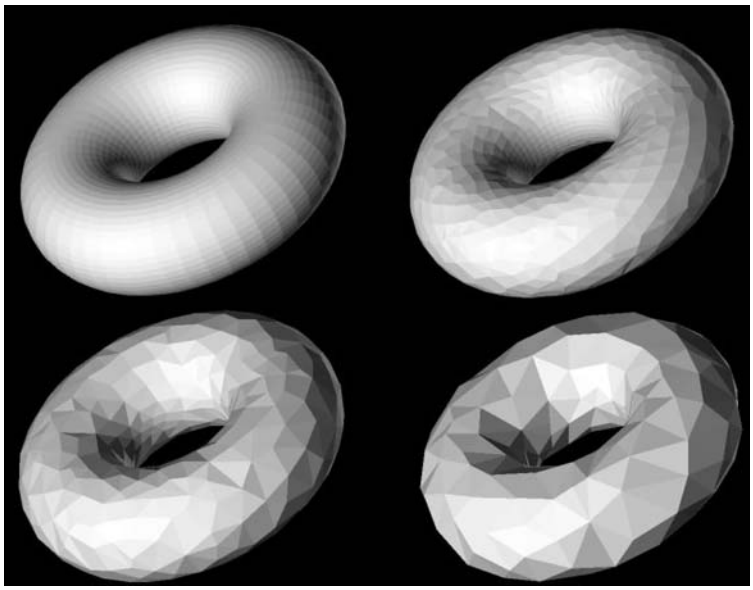

Fig. 5.10. Triangle reduction of 50%, 80%, and 90% for the torus $\big(\ (2+\cos u)\cos v, (2+\cos u)\sin v, \sin u\ \big)^T$, $u, v \in [0, 2\pi]$.

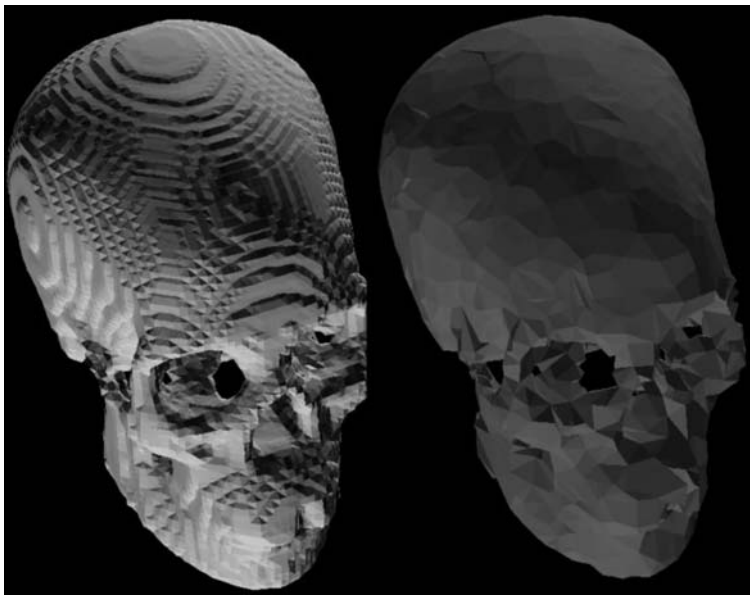

Fig. 5.11. Triangle reduction of 90% for a piecewise triangular approximation of a human skull.

The triangle reduction strategy is tested for graphs of the same bivariate functions as used in chapter 4.2., Table 4.1. Again, all test functions $f(x, y)$ are defined over $[-1, 1] \times [-1, 1]$ and evaluated on a $51 \cdot 51-$grid with equidistant spacing,

$$(x_i, y_j)^T \;=\; \Big(-1 + \frac{i}{25}, -1 + \frac{j}{25} \Big)^T, \qquad i, j = 0...50,$$

determining points on their graphs,

$$\Big\{ \big(\; x_i, y_j, f(x_i, y_j) \; \big)^T \mid i, j = 0...50 \Big\}.$$

A graph's original triangulation is obtained by splitting each quadrilateral specified by its index quadruple

$$\big(\; (i, j), (i + 1, j), (i + 1, j + 1), (i, j + 1) \; \big)$$

into the two triangles $T^1_{i,j}$ and $T^2_{i,j}$ identified by their index triples,

$$T^1_{i,j} \;=\; \big(\, (i,j), (i{+}1, j), (i{+}1, j{+}1) \, \big) \qquad \text{and} \qquad T^2_{i,j} \;=\; \big(\, (i,j), (i{+}1, j{+}1), (i, j{+}1) \, \big).$$

The initial triangulation is now reduced using the new technique. Determining a piecewise linear function, original and reduced triangulation are compared at the given knots, (x_i, y_j), $i, j = 0...50$. Therefore, the root-mean-square error is

$$\sqrt{\frac{1}{51} \; \frac{1}{51} \sum_{j=0}^{50} \sum_{i=0}^{50} \big(f(x_i, y_j) - \hat{f}(x_i, y_j) \big)^2}, \tag{5.41.}$$

where $\hat{f}$ denotes the piecewise linear function implied by the reduced triangulation.

In the next table, original and reduced triangulation are compared for different reduction rates, i.e., the original number of triangles is reduced by 50%, 80%,

and 90%. Newly constructed triangles can not be removed from an intermediate triangulation in the next iteration step, unless there is no other choice. During the reduction process it is ensured that the final reduced triangulation still covers the whole domain $[-1,1] \times [-1,1]$.

Tab. 5.1. RMS errors of triangle reduction for graphs of bivariate functions.

Function	50%	80%	90%
1. Plane: $.2\ (x+y)$.	0	0	0
2. Cylinder: $\sqrt{2-x^2}$.	.000485	.000999	.002105
3. Sphere: $\sqrt{4-(x^2+y^2)}$.	.000445	.001234	.002288
4. Paraboloid: $.4\ (x^2+y^2)$.	.000610	.001591	.003619
5. Hyperboloid: $.4\ (x^2-y^2)$.	.000248	.000794	.002009
6. Monkey saddle: $.2\ (x^3-3xy^2)$.	.000355	.000843	.001853
7. Cubic polynomial: $.15\ (x^3+2x^2y-xy+2y^2)$.	.000381	.001029	.002160
8. Exponential function: $e^{-\frac{1}{2}(x^2+y^2)}$.	.000336	.000916	.002075
9. Trigonometric function: $.1\ \big(\cos(\pi x)+\cos(\pi y)\big)$.	.000359	.001088	.002054

Chapter 6

A triangular tangent-plane-continuous surface

6.1. Introduction

In CAD applications one is often concerned with the problem of generating smooth surfaces being unions of triangular patches, e.g., car bodies. For most purposes, at least tangent-plane continuity is required between adjacent patches. Commonly, the given data are points and associated outward normal vectors (or pairs of tangent vectors) to be interpolated in three-dimensional space. A triangulation of the data points must be known as well.

Triangular methods for the function setting, where points in the plane and associated function and derivative values are given, are discussed in [Barnhill et al.'73], [Barnhill & Farin '81], and [Farin '86]. The more general problem of interpolating arbitrary points in three-dimensional space with prescribed tangent planes has been considered later, e.g., in [Farin '83]. In [Nielson '87] and [Hamann et al.'90] a tangent-plane-continuous surfaces are constructed based on a so-called side-vertex method, originally introduced for bivariate functions (see [Nielson '79]).

Other methods are described in [Herron '85] and [Piper '87]. In [Hagen '89] tangent planes as well as principal curvatures at the data points are interpolated yielding a G^2 surface. A completely different approach is chosen in [Sederberg '85] and [Dahmen '89], where surface patches are implicitly defined by trivariate functions.

The surface scheme developed here is based on the side-vertex technique and

on degree elevated conics as the underlying curve scheme, since the given data may imply curves with and without inflection points. The conic scheme is modified in order to allow more general curves. Ball's generalized conics can be used as an alternative (see [Ball '74,'75,'77] and [Boehm '82]).

In principle, the problem is to interpolate points $\mathbf{x}_i \in I\!R^3$ with given outward (unit) normal vectors $\mathbf{n}_i$. A two-dimensional triangulation $\mathcal{T}$ defining the vertices in $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ forming triangles must be known for the data points. Patch boundary curves are first constructed along the edges of each triangle, then, a radial projector is used to blend from a vertex $\mathbf{v}_i$ to the opposite boundary curve along edge e_i, $i = 1, 2, 3$. The curves used for this blending process are degree elevated conics, being modified such that both convex and rational cubic curves with an inflection point can be generated.

Boundary curves are generated first. Then, three patch building blocks are obtained by calculating degree elevated conics, emanating from a triangle vertex and ending at a point on the opposite boundary curve. Finally, the patch building blocks are blended together in a convex combination defining the complete patch.

The (intrinsic) domain for each patch is the set of triples (u_1, u_2, u_3) of barycentric coordinates for which $\sum_{i=1}^{3} u_i = 1$, $u_i \geq 0$, $i = 1, 2, 3$. Each point on a patch is the image of a triple (u_1, u_2, u_3).

Definition 6.1. The convex combination

$$\mathbf{s}\ (\mathbf{u}) = \sum_{i=1}^{3}\ w_i(\mathbf{u})\ \mathbf{s}_i(\mathbf{u}) \tag{6.1.}$$

interpolating the triangle vertices $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ and associated outward (unit) normal vectors $\mathbf{n}_1$, $\mathbf{n}_2$, and $\mathbf{n}_3$, where $\mathbf{u} = (u_1, u_2, u_3)$ represents the barycentric coordinates of a point in the triangle and $\sum_{i=1}^{3} \omega_i(\mathbf{u}) = 1$, $\omega_i(\mathbf{u}) \geq 0$, is called a **six parameter patch**.

Each building block $\mathbf{s}_i(\mathbf{u})$ of the patch interpolates the positional data along all three triangle edges and the normal data along the opposite edge e_i. The final patch $\mathbf{s}(\mathbf{u})$ interpolates the positional and normal data prescribed along all three edges. A particular set of weight functions ω_i is needed to solve the interpolation problem.

Theorem 6.1. *The weight functions*

$$w_i\ (\mathbf{u}) \ = \ \frac{B^2_{(1,1,1)-e_i}(\mathbf{u})}{B^2_{(0,1,1)}(\mathbf{u}) + B^2_{(1,0,1)}(\mathbf{u}) + B^2_{(1,1,0)}(\mathbf{u})}, \qquad i = 1, 2, 3, \tag{6.2.}$$

where $\mathbf{e}_1 = (1, 0, 0)$, $\mathbf{e}_2 = (0, 1, 0)$, *and* $\mathbf{e}_3 = (0, 0, 1)$, *and* $B^2_{(i,j,k)}$, $i + j + k = 2$, *are the Bernstein polynomials of degree two defined as*

$$B^2_{(i,j,k)}\ (\mathbf{u}) \ = \ \frac{2}{i!\ j!\ k!}\ u_1^i\ u_2^j\ u_3^k, \tag{6.3.}$$

have the properties

(i) $\displaystyle\sum_{i=1}^{3} w_i(\mathbf{u}) \ = \ 1,$

(ii) $w_i\ (e_k) \ = \ \delta_{i,k}, \quad i, k \in \{1, 2, 3\}, \quad$ *and*

(iii) $D_{\mathbf{d}}\ \big(w_i(e_i)\big) \ = \ 0, \quad i \in \{1, 2, 3\},$

where edge e_1 *is characterized by barycentric coordinates* $(0, u_2, u_3)$, *edge* e_2 *by* $(u_1, 0, u_3)$, *and edge* e_3 *by* $(u_1, u_2, 0)$, $\delta_{i,k}$ *is the Kronecker delta, and* $D_{\mathbf{d}}$ *is a direc-*

tional derivative in any direction $\mathbf{d}$, *where* $\mathbf{d}$ *is expressed in barycentric coordinates* (d_1, d_2, d_3), $\sum_{i=1}^{3} d_i = 0$.

Proof. See [Nielson '79].

Definition 6.2. The single **patch building blocks** $\mathbf{s}_i(\mathbf{u})$, $i = 1, 2, 3$, are called **compatible** if each interpolates all three boundary curves of the triangular patch $\mathbf{s}(\mathbf{u})$ and the normals edge e_i, formalized

$$\text{(iv)} \quad \mathbf{s}_i[\mathbf{c}]\ (e_j) \ = \ \mathbf{c}\ (e_j), \qquad i = 1, 2, 3,\ j \in \{1, 2, 3\} \quad \text{and}$$

$$\text{(v)} \quad \mathbf{n}\big[\mathbf{s}_i[\mathbf{c}]\big]\ (e_i) \ = \ \mathbf{n}[\mathbf{c}]\ (e_i), \quad i = 1, 2, 3.$$

A boundary curve $\mathbf{c}$, the patch building blocks $\mathbf{s}_i$, and the normal $\mathbf{n}$ are viewed as operators. The notation "[ ]" means "restricted to." Using the properties (i), (ii), and (iii) from Theorem 6.1. and (iv) and (v) from Definition 6.2., the following interpolation theorem holds.

Theorem 6.2. *The convex combination*

$$\mathbf{s}\ (\mathbf{u}) \ = \ \sum_{i=1}^{3} w_i(\mathbf{u})\ \mathbf{s_i}(\mathbf{u}) \tag{6.4.}$$

interpolates all three boundary curves and the patch normals on the boundary.

Proof. a) Positional interpolation:

It is $\mathbf{s}_i[\mathbf{c}]\ (e_j) = \mathbf{c}\ (e_j)$, $i = 1, 2, 3$, $j \in \{1, 2, 3\}$, and $\sum_{i=1}^{3} w_i(e_j) = 1$. Therefore, $\mathbf{s}[\mathbf{c}]\ (e_j) = \mathbf{c}\ (e_j)$ holds.

b) Normal interpolation:

To show interpolation of the boundary normals one calculates two non-parallel

tangent vectors for a point on a boundary curve, calculates the cross product and shows that it coincides with the prescribed patch normal at that point. Let the directions in which tangent vectors are computed be $\mathbf{d}_1 = (-1, 1, 0)$, $\mathbf{d}_2 = (0, -1, 1)$, and $\mathbf{d}_3 = (1, 0, -1)$. $\mathbf{D}_{\mathbf{d}_i}$ is the vector valued derivative operator determining tangent vectors in direction $\mathbf{d}_i$. Using the product rule and taking the properties (ii) and (iv) into account (Theorem 6.1., Definition 6.2.), one obtains

$$\mathbf{D}_{\mathbf{d}_i}\mathbf{s}[\mathbf{c}]\ (e_i)$$

$$= \ w_1(e_i)\ \mathbf{D}_{\mathbf{d}_i}\mathbf{s}_1[\mathbf{c}]\ (e_i) + D_{\mathbf{d}_i}w_1(e_i)\ \mathbf{s}_1[\mathbf{c}]\ (e_i)$$

$$+w_2(e_i)\ \mathbf{D}_{\mathbf{d}_i}\mathbf{s}_2[\mathbf{c}]\ (e_i) + D_{\mathbf{d}_i}w_2(e_i)\ \mathbf{s}_2[\mathbf{c}]\ (e_i)$$

$$+w_3(e_i)\ \mathbf{D}_{\mathbf{d}_i}\mathbf{s}_3[\mathbf{c}]\ (e_i) + D_{\mathbf{d}_i}w_3(e_i)\ \mathbf{s}_3[\mathbf{c}]\ (e_i)$$

$$= \ \mathbf{D}_{\mathbf{d}_i}\mathbf{s}_i[\mathbf{c}]\ (e_i) + \mathbf{c}\ (e_i)\ \Big(D_{\mathbf{d}_i}\big(w_1(e_i) + w_2(e_i) + w_3(e_i)\big)\Big)$$

$$= \ \mathbf{D}_{\mathbf{d}_i}\mathbf{s}_i[\mathbf{c}]\ (e_i).$$

Considering the result from a), the tangent vector along a boundary curve is given by

$$\mathbf{D}_{\mathbf{d}_{1+(i \bmod 3)}}\mathbf{s}_i[\mathbf{c}]\ (e_i) \ = \ \mathbf{D}_{\mathbf{d}_{1+(i \bmod 3)}}\mathbf{s}[\mathbf{c}]\ (e_i),$$

$i = 1, 2, 3$. Choosing two arbitrary directions $\mathbf{d}_i$ and $\mathbf{d}_{1+(i \bmod 3)}$, $i = 1, 2, 3$, the patch normal along a boundary is determined by the cross product

$$\mathbf{n}[\mathbf{c}]\ (e_i) \ = \ \mathbf{n}\big[\mathbf{s}_i[\mathbf{c}]\big]\ (e_i)$$

$$= \ \mathbf{D}_{\mathbf{d}_i}\mathbf{s}_i[\mathbf{c}]\ (e_i) \times \mathbf{D}_{\mathbf{d}_{1+(i \bmod 3)}}\mathbf{s}_i[\mathbf{c}]\ (e_i) \ = \ \mathbf{D}_{\mathbf{d}_i}\mathbf{s}[\mathbf{c}]\ (e_i) \times \mathbf{D}_{\mathbf{d}_{1+(i \bmod 3)}}\mathbf{s}[\mathbf{c}]\ (e_i)$$

$$= \mathbf{n}\big[\mathbf{s}[\mathbf{c}]\big]\ (e_i),$$

proving normal interpolation along the boundaries.

q.e.d.

The concept of barycentric coordinates for a triangle is shown in Figure 6.1.

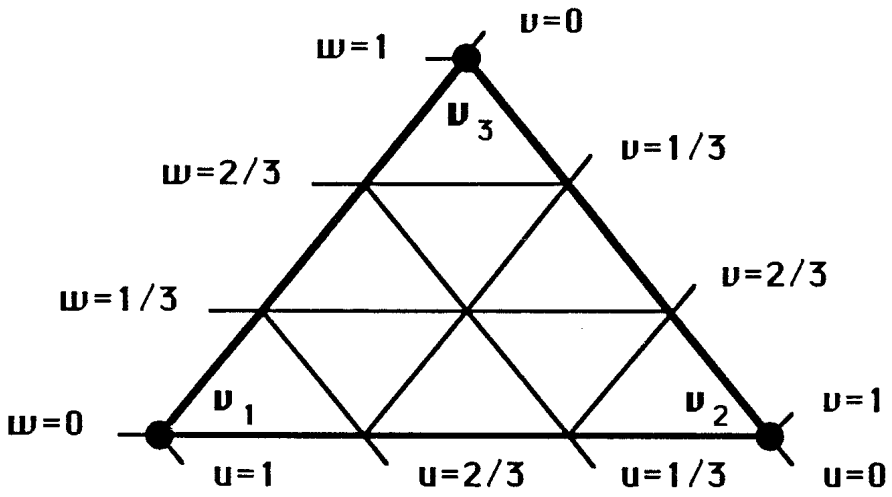

Fig. 6.1. Concept of barycentric coordinates for a triangle.

6.2. The conic curve scheme

A planar curve scheme is needed for the interpolation of two points $\mathbf{b}_0 \in I\!R^3$ and $\mathbf{b}_3 \in I\!R^3$ and two associated outward unit normal vectors, $\mathbf{n}_0 \in I\!R^3$ and $\mathbf{n}_3 \in I\!R^3$. A plane containing unit tangent vectors for the end points of the curve must be defined.

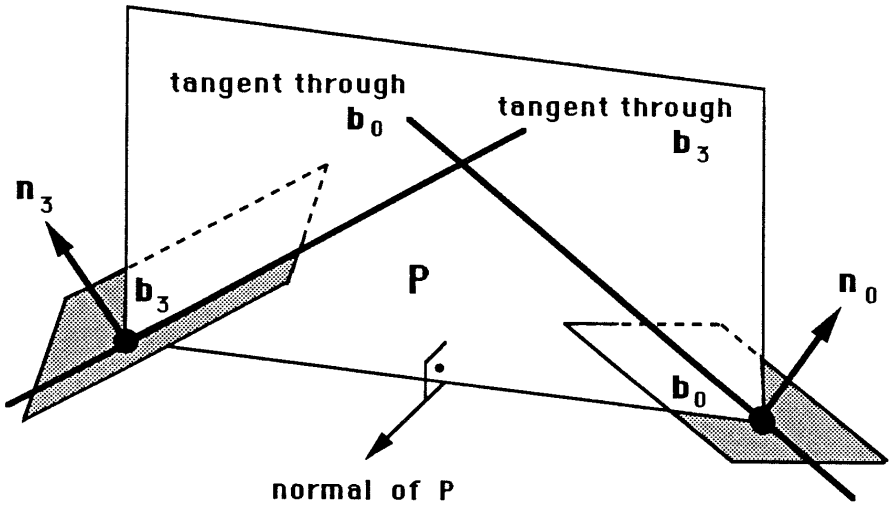

Fig. 6.2. Conic in Bézier representation.

Referring to Figures 6.2. and 6.3., the construction proceeds as follows:

(i) Define a plane P through $\mathbf{b}_0$ and $\mathbf{b}_3$ containing the desired curve. This plane is specified by the requirement that the vector $\frac{1}{2}(\mathbf{n}_0+\mathbf{n}_3)$ lies in it (special care necessary for the case $\mathbf{n}_0 = -\mathbf{n}_3$).

(ii) Construct the intersection of the conic plane P and the tangent plane P_0 at $\mathbf{b}_0$ and of P and the tangent plane P_3 at $\mathbf{b}_3$. Each of the straight lines obtained defines the tangent of the desired curve at $\mathbf{b}_0$ and $\mathbf{b}_3$, respectively. The cross product between the normal of P and the two given normal vectors at the end points of the curve define unit tangent vectors for the end points, denoted by $\mathbf{t}_0$ and $\mathbf{t}_3$.

(iii) The resulting conic is written as a degree elevated rational Bézier curve of degree three,

$$\mathbf{c}\ (t)\ =\ \frac{\sum\limits_{i=0}^{3}\ \omega_i\ \mathbf{b}_i\ B_i^3(t)}{\sum\limits_{i=0}^{3}\ \omega_i\ B_i^3(t)}, \tag{6.5.}$$

where $t \in [0,1]$, $\omega_0 = \omega_3 = 1$, $\omega_1 = \omega_2 = \omega$, and

$$B_i^3\ (t)\ =\ \binom{3}{i}\ (1-t)^{3-i}\ t^i, \qquad i = 0...3. \tag{6.6.}$$

Referring to Figure 6.3., the interior Bézier points, $\mathbf{b}_1$ and $\mathbf{b}_2$, lie on a line parallel to the line through $\mathbf{b}_0$ and $\mathbf{b}_3$. Using the law of sines one obtains

$$l_{0,1}\ =\ \frac{\sin\beta}{\sin\gamma}\ L_{0,3}. \tag{6.7.}$$

Degree elevation requires the following ratio to hold (see [Farin '90]):

$$\frac{L_{0,1}}{l_{0,1} - L_{0,1}}\ =\ 2\ \omega. \tag{6.8.}$$

Therefore,

$$L_{0,1}\ =\ \frac{2\ \omega}{1+2\ \omega}\ \frac{\sin\beta}{\sin\gamma}\ L_{0,3}. \tag{6.9.}$$

Thus, one gets

$$\mathbf{b}_1\ =\ \mathbf{b}_0 + L_{0,1}\mathbf{t}_0. \tag{6.10.}$$

The same construction is carried out for $\mathbf{b}_2$.

Theorem 6.3. *Choosing the weight* $\omega = \omega_1 = \omega_2$ *as*

$$\omega\ =\ \sin\frac{\gamma}{2}\ =\ \cos\alpha \tag{6.11.}$$

determines a finite value $L_{0,1}$ for γ approaching zero (parallel tangents at end points) and defines a circular arc for the case $\alpha = \beta$.

Proof. Obviously, the scheme yields circular arcs for an isosceles triangle as Bézier polygon (see [Boehm et al.'84]).

The choice for ω also guarantees a finite value for $L_{0,1}$, since

$$\begin{aligned} \lim_{\gamma\to 0} L_{0,1} &= \lim_{\gamma\to 0} \frac{2\ \sin\frac{\gamma}{2}}{1+2\ \sin\frac{\gamma}{2}} \frac{\sin\beta}{\sin\gamma} L_{0,3} \\ &= \lim_{\gamma\to 0} \frac{2\ \frac{\gamma}{2}}{1+2\ \frac{\gamma}{2}} \frac{\sin\beta}{\gamma} L_{0,3} = L_{0,3}\ \sin\beta. \end{aligned} \tag{6.12.}$$

q.e.d.

In order to avoid consistency problems between patches and to reduce input information all weights associated with interior Bézier points should be chosen automatically as proposed in Theorem 6.3. Of course, they can also be specified by the user, considering the consistency constraints. Figure 6.3. illustrates the degree elevation process for conics.

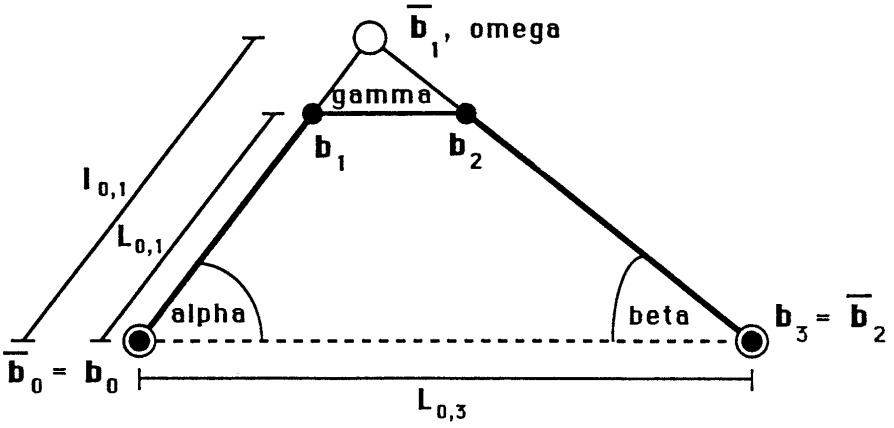

Fig. 6.3. Degree elevation for a conic in Bézier representation.

6.3. Computing the patch building blocks

The computation of a point $\mathbf{s}_i(\mathbf{u})$, $i = 1, 2, 3$, for given parameter values u_1, u_2, and u_3 is based on generating two separate curves. The first curve is associated with the edge e_i of the domain triangle interpolating the vertices associated with this edge and the computed tangent vectors at those vertices.

This curve is evaluated to obtain a point on the boundary along edge e_i. The second curve constructed is the result of blending from the vertex $\mathbf{v}_i$ to the point on the curve along e_i, thus interpolating the vertex $\mathbf{v}_i$, the point on the curve along e_i, and the two tangent vectors prescribed for these two points. This curve finally determines a point on this building block.

The planar conic scheme is used for the computation of points on a building block $\mathbf{s}_i(\mathbf{u})$ in the following way:

(i) *Curve scheme for the boundary curves*

Using the planar curve scheme based on degree elevated conics it is easy to generate the three boundary curves $\mathbf{c}_1(t)$, $\mathbf{c}_2(t)$, and $\mathbf{c}_3(t)$. For the computation of patch building block $\mathbf{s}_1(\mathbf{u})$ at $\mathbf{u} = (u_1, u_2, u_3)$ one evaluates the curve $\mathbf{c}_1(t)$ for $t = u_3/(1 - u_1) \in [0, 1]$ along edge e_1. The input for the conic scheme are the vertices $\mathbf{v}_2$ and $\mathbf{v}_3$, the normals $\mathbf{n}_2$ and $\mathbf{n}_3$, and the parameter t. This results in a particular point on the patch boundary.

(ii) *Surface scheme for a point on* $\mathbf{s}_i(\mathbf{u})$

Having computed a point $\mathbf{c}_i(t)$, $i = 1, 2, 3$, on a boundary curve, the next step is the estimation of the surface normal of the final patch at a particular point

on $\mathbf{c}_i(t)$. The surface normal $\mathbf{n}_i^S(t)$ along $\mathbf{c}_i(t)$ must be perpendicular to this curve itself,

$$\mathbf{n}_i^S(t) \cdot \dot{\mathbf{c}}_i(t) = 0, \tag{6.13.}$$

$i = 1, 2, 3$, where $\dot{\mathbf{c}}_i(t)$ denotes the tangent vector of the conic. Requiring

$$\mathbf{n}_i^S(t) \cdot \mathbf{n}_i^C(t) = \gamma\,(t), \tag{6.14.}$$

where $\mathbf{n}_i^C(t)$ denotes the unit normal vector to the conic in its plane, $\mathbf{n}_i^S(t)$ is determined. The value $\gamma\,(t)$ is chosen according to the following interpretation: At $t = 0$,

$$\mathbf{n}_i^S(0) \cdot \mathbf{n}_i^C(0) = \gamma\,(0) \tag{6.15.}$$

denotes the cosine of the angle formed by the surface normal $\mathbf{n}_i^S(0)$ and the conic normal $\mathbf{n}_i^C(0)$. At $t = 1$, $\gamma\,(1)$ has the analogous interpretation. If one sets

$$\gamma\,(t) = (1-t)\,\gamma(0) + t\,\gamma(1), \tag{6.16.}$$

the cosine of the angle formed by $\mathbf{n}_i^S(t)$ and $\mathbf{n}_i^C(t)$ varies linearly along the edge.

This process guarantees that the surface normals are the same as the given ones at the vertices and, moreover, that the surface normals along an edge are the same for this triangular surface patch as well as for a neighbor patch sharing the edge. Thus, tangent-plane continuity between adjacent patches is assured.

Remark 6.1. The choice of the boundary surface normal considers the boundary curve itself, a simple linear interpolation of the given normal vectors at the two vertices is avoided.

The process of determining the surface normal along the boundary conic $\mathbf{c}_1(t)$ is illustrated in Figure 6.4.

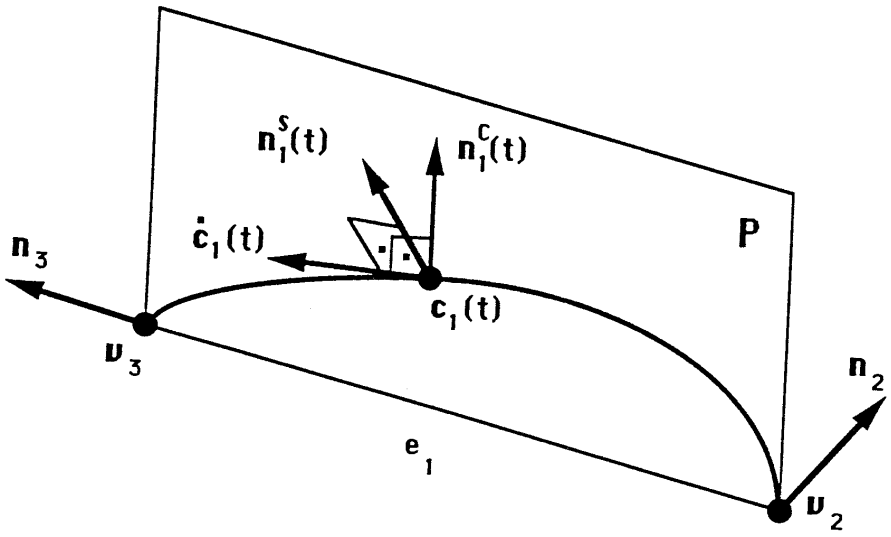

Fig. 6.4. Generating patch normal along edge e_1.

Using the idea of radial projectors, the blending from a vertex to the opposite boundary curve is done next. This is described for the vertex $\mathbf{v}_1$ and its associated boundary curve $\mathbf{c}_1(t)$. The generation of a point on a curve, emanating from $\mathbf{v}_1$ and ending at a point on $\mathbf{c}_1(t)$, follows the same principle as the generation of a point on the boundary conic $\mathbf{c}_1(t)$, $t = u_3/(1-u_1) \in [0,1]$.

To obtain a point on the patch building block $\mathbf{s}_1(\mathbf{u})$ at $\mathbf{u} = (u_1, u_2, u_3)$ one has to construct a curve $\mathbf{c}(\overline{t})$, $\overline{t} = (1 - u_1) \in [0, 1]$. The input data for the curve scheme are the vertices $\mathbf{v}_1$ and $\mathbf{c}_1(t)$, the normals $\mathbf{n}_1$ and the constructed surface normal $\mathbf{n}_1^S(t)$ along edge e_1. The computation of a point on $\mathbf{s}_1(\mathbf{u})$ is shown in Figure 6.5.

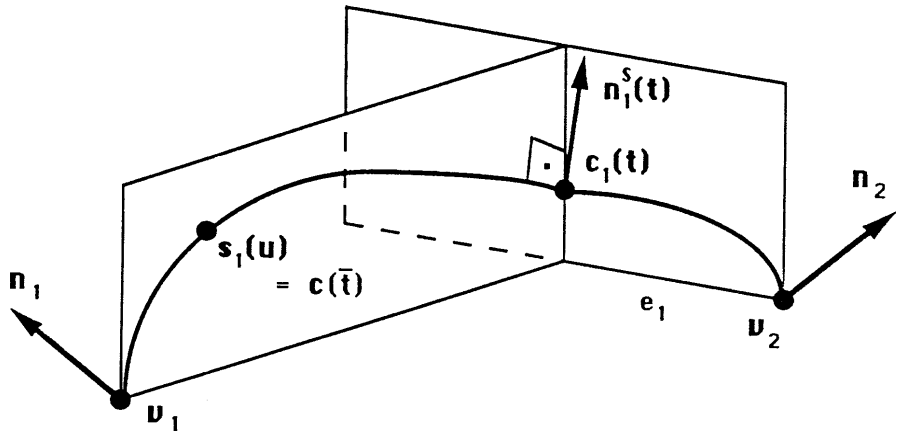

Fig. 6.5. Evaluating first patch building block.

Repeating this process for the other two building blocks $\mathbf{s}_2(\mathbf{u})$ and $\mathbf{s}_3(\mathbf{u})$ finally yields the point $\mathbf{s}(\mathbf{u})$ on the surface. The weights for the interior Bézier points of all conics can be interpreted as tension parameters, allowing the generation of patches approaching the triangle $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ for small weights ($\omega \ll 1$).

So far, only convex data configurations have been considered, making it possible to use conics. Generalized conics were introduced in [Ball '74,'75,'77] as rational curves of degree three, including conics as a subset. The concept of generalized

conics allows to model input data implying curves with and without an inflection point. A criterion must be given that allows to decide, whether the presented conic scheme can be used for a planar data configuration.

Definition 6.3. The line through the vertices $\mathbf{v}_1$ and $\mathbf{v}_2$ divides the plane into two half-planes. The tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ associated with $\mathbf{v}_1$ and $\mathbf{v}_2$ define a **convex configuration** if the tangent vectors are directed into opposite half-planes and a **non-convex configuration**, otherwise.

If given data are convex the planar scheme for degree elevated conics can be used as described above. In the case of a non-convex configuration, the two interior Bézier points for the curve scheme must be constructed in a way that a rational curve of degree three with an inflection point is obtained.

Assuming the tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ are directed into the same half-plane, the prescribed tangents through $\mathbf{v}_1$ and $\mathbf{v}_2$ are reflected with respect to the axis given by the line through $\mathbf{v}_1$ and $\mathbf{v}_2$. Therefore, two pairs of tangents are obtained, one pair per vertex. The degree elevation procedure is then carried out in both half-planes. The interior Bézier points must be chosen in a way such that $(\mathbf{b}_1 - \mathbf{b}_0) = \alpha_1 \mathbf{t}_1,\ \alpha_1 > 0$, and $(\mathbf{b}_3 - \mathbf{b}_2) = \alpha_2 \mathbf{t}_2,\ \alpha_2 > 0$.

The four different data configurations possible are illustrated in Figure 6.6. Two curves with and two without inflection points can be obtained.

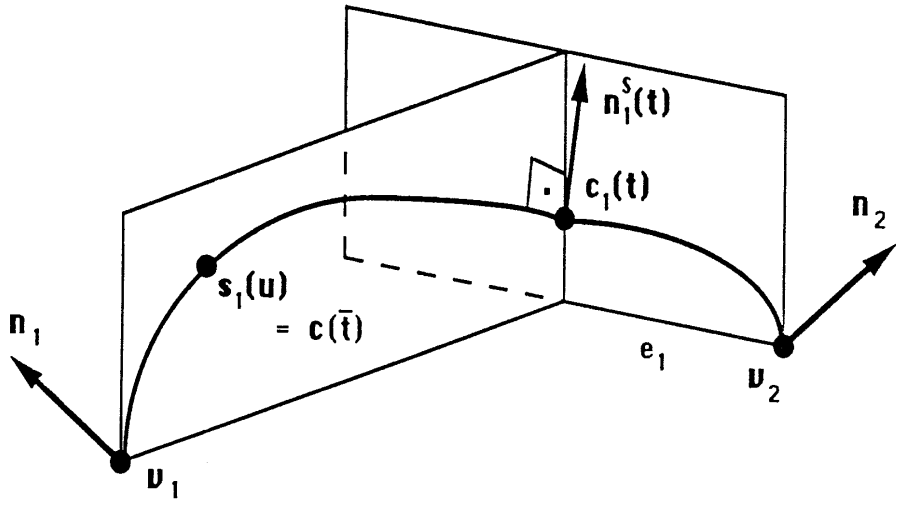

Fig. 6.6. Convex and non-convex data configurations defined by end points and end tangents in a plane.

Thus, the concept of degree elevated conics allows us to handle both convex and non-convex data. The continuity inside a single patch building block is guaranteed, since the construction of the interior Bézier points is continuous with respect to the involved angles.

Remark 6.2. If all data points $\mathbf{x}_j,\ j \neq i$, are lying in the same half-space determined by the plane through $\mathbf{x}_i$ with normal $\mathbf{n}_i$, and this is true for all i, a convex surface is implied and "usual" conics can be used everywhere as curve scheme.

Remark 6.3. Choosing the weights automatically, as described in Theorem 6.3., yields circular arcs when the data configuration implies this. Choosing points and normals from a unit sphere as input data produces a surface rather well approximating a sphere. However, the presented scheme does not have spherical precision. This

is the result of the convex combination. Each single building block has spherical precision, but each one generates a different point on the sphere.

Remark 6.4. Using degree elevated conics instead of parametric cubic curves guarantees that one does not obtain inflection points or loops unless the prescribed normals at the two end points of a curve imply an inflection point.

In Figure 6.7., three different surfaces are shown using increasing weights. The four vertices of an equilateral tetrahedron inscribed in a unit sphere with the associated normals of the sphere at these points are given as input. For the first surface the weight ω is chosen automatically (Theorem 6.3.). In this case, the maximal distance of the resulting surface from the unit sphere is about 0.01. The other two surfaces both have lower weights.

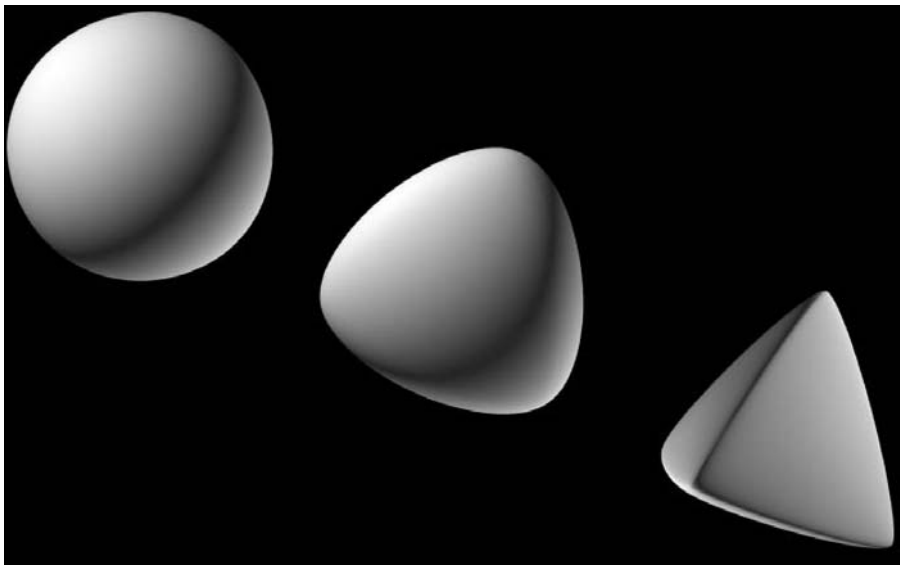

Fig. 6.7. Triangular surfaces obtained from spherical data using increasing weights (from left to right).

In Figure 6.8., the surface scheme has been applied to the reduced triangulation approximating a human skull (about 6,000 triangles, see Figure 5.11.). All weights ω are chosen automatically (Theorem 6.3.).

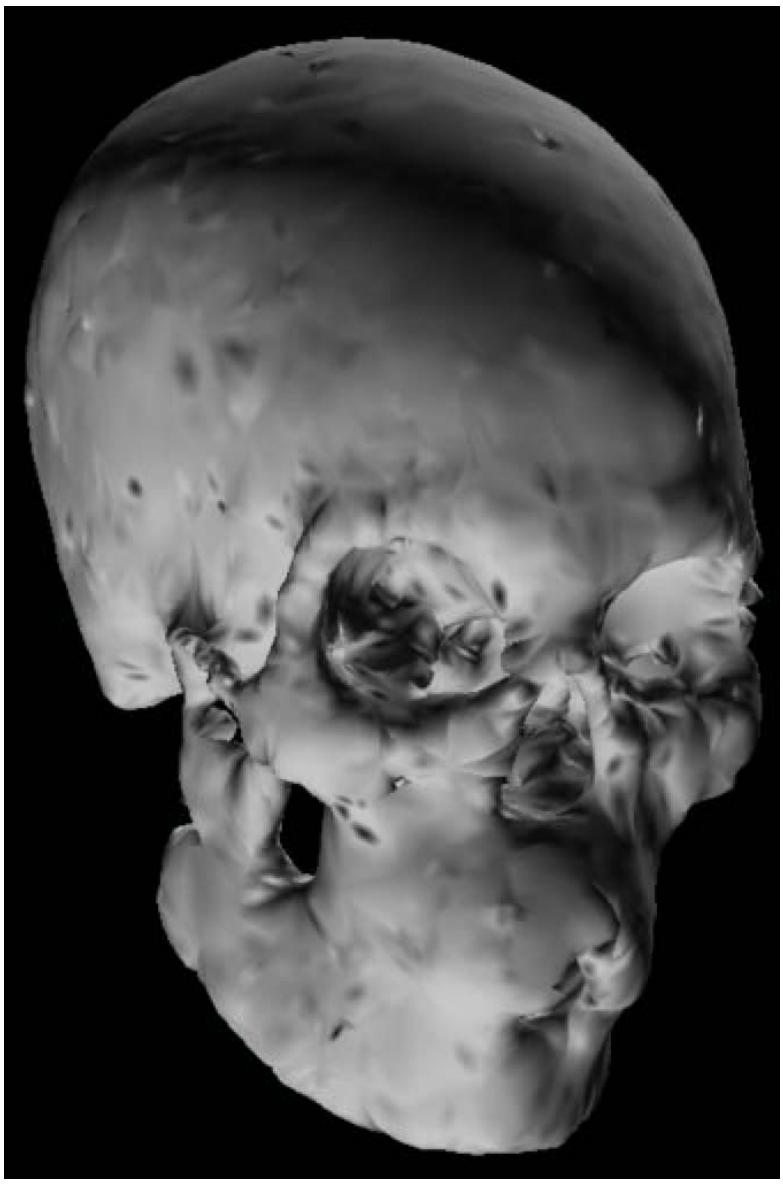

Fig. 6.8. Triangular surface for reduced skull triangulation, weights chosen automatically.

Chapter 7

Conclusions

The dissertation has presented several ideas for the visualization of trivariate data. More research in this area is necessary, since computing power becomes more and more accessible, while visualization techniques, particularly for dynamical systems, are in a rather primitive state.

A new approach has been introduced for visualizing and modeling trivariate data (scalar fields). Following this approach, a contour of some trivariate function or a finite, discrete trivariate data set is approximated first and then used for modeling. This process itself can be seen as data reduction, since the contour approximation leads to a two-dimensional triangulation. This strategy might not be applicable for all real-world problems, but in the case of discontinuous scalar fields, e.g., CAT scan data, it is definitely an alternative to the standard way of constructing a trivariate interpolant for all data.

An existing technique for computing a triangular approximation to a contour of a trivariate function, the so-called *marching-cubes method*, has been corrected and improved. The contour approximation produces a continuous triangulation for which additional topological information (neighbors of triangles) is generated. Continuity of a triangulation is necessary for further modeling.

A way for approximating the two principal curvatures at the vertices in a two-dimensional triangulation in three-dimensional space has been developed and extended to the approximation of the three principal curvatures at the vertices on the three-dimensional graph of a trivariate function in four-dimensional space. This leads to a method for analyzing the smoothness of two-dimensional surfaces and of hypergraphs of trivariate functions, e.g., trivariate interpolants and approximants. Approximation schemes requiring curvature input can make use of the principal curvature approximation as well.

Most data reduction techniques can only be applied to function data, e.g., to sets of points in the plane (or space) with function values. The new triangle removal algorithm can be used for general two-dimensional triangulations in three-dimensional space. A triangulation is adaptively reduced such that at each reduction step the implied piecewise triangular approximant of the surface changes as little as possible. The same strategy could also be applied to the removal of tetrahedra in a tetrahedrization of points in three-dimensional space (with function values), taking the absolute curvature at points on the implied piecewise linear hypergraph in four-dimensional space into account.

An elegant planar curve scheme based on degree elevated conics has been developed. It is utilized as a blending technique, needed for a particular triangular surface scheme, the side-vertex method. Combining the side-vertex method with the new curve scheme results in smooth surfaces. The triangular, tangent-plane

continuous surface can be viewed as an alternative to existing triangular interpolants for general two-dimensional triangulations in three-dimensional space.

Biographical Sketch

Name:	Bernd Hamann
Date of birth:	24 December 1963, Alfeld, Germany
Academic education:	Doctor of Philosophy, 1991, Arizona State University, U.S.A.
	Master of Science in computer science, 1988, Technical University Braunschweig, Germany
	Bachelor of Science in mathematics, 1986, Technical University Braunschweig
	Bachelor of Science in computer science, 1985, Technical University Braunschweig
Practical experience:	Application programmer, 1985-1986, Stahlwerke Peine-Salzgitter AG, Germany
	Research/teaching assistant, 1988-1991, computer science, Arizona State University
Research interests:	Rendering and visualization techniques for multivariate scientific data (e.g., medical imaging, meteorology, geology, geography)
	Development of curve and surface techniques for design purposes and data approximation (e.g., car bodies, ship hulls, airplanes)
	In general: computer graphics, computer-aided geometric design (CAGD), numerical mathematics, computer vision, robotics