Arbeitsbereich Technische Informatik I

D I P L O M A R B E I T

Sven Utcke

Transfer and Invariants of Surfaces of Revolution

Erstellt in Zusammenarbeit mit

UNIVERSITY OF OXFORD

Department of Engineering Science

Technische Universität Hamburg-Harburg
Technische Informatik I
Prof. Dr.-Ing. Hans Burkhardt

Harburger Schloßstr. 20
2100 Hamburg 90
Tel. 040-7718-3025

February 7, 2001

Aufgabenstellung für die Diplomarbeit
von Herrn Sven Utcke

Transfer and Invariants of Surfaces of Revolution

0.1 Introduction

A number of recent papers in the Computer Vision and Pattern Recognition literature have demonstrated that invariants, or equivalently structure modulo a 3D linear transformation, are sufficient for object recognition [1, 19, 20]. The final stage in the recognition process is verification, where an outline is *transferred* from an acquisition image of the object to the target image.

For the most part recognition based on invariants has concentrated on planar objects [19], though recently 3D invariants have been measured from single and multiple images for polyhedra [18], point sets [9, 13, 18], surfaces of revolution [12] and algebraic surfaces [11]. The work so far on surfaces of revolution has only exploited isolated points on the outline (such as bitangents), and has not addressed transfer or verification.

The aim of this project is to extend the transfer and extraction of invariants to surfaces of revolution using the entire outline.

0.2 Specification

A surface of revolution is simply a rotated *generating curve*. There are two related goals for the project:

1. **Transfer:** Given a single (or multiple) views of the surface, obtain the projection in any other given view. For example, after specifying a

minimal number of points in the target image, render the object from that viewpoint.

2. **Invariants:** Extract from the outline in a single view a *signature* or set of invariants which are viewpoint independent. These should also be derived directly from the generating curve.

The project will be developed in a number of stages. In the first place the affine approximation to projection will be employed (so that the object is imaged under parallel projection). This has the virtue that the essential geometry of the contour generator depends only on one parameter - the direction of projection. The second stage will investigate perspective projection, where (two) parameters specifying the optical center must be considered.

The analysis will be partly theoretical - employing the symbolic algebra package Mathematica, and partly experimental. The goal is to develop methods that work reliably and robustly on images of real objects.

0.3 Relevant Literature

Papers listed below on the application of invariants to model based recognition. Also background texts on projective geometry [16, 22, 24] and differential geometry [8, 15, 21].

Referent:	Prof. Dr.-Ing. H. Burkhardt
Betreuer:	Dr. A. Zisserman (University of Oxford)
Ausgabedatum:	01.08.1993
Abgabedatum:	31.01.1993
Bearbeitungszeit:	6 Monate

.....
Prof. Dr.-Ing. H. Burkhardt

This thesis is entirely my own work and, except where otherwise stated, describes my own research.

Oxford, February 7, 2001

Contents

Chapter 1

Introduction

"Begin at the beginning," the King said, gravely, "and go on till you come to the end: then stop."

Lewis Carroll, *Alice in Wonderland*

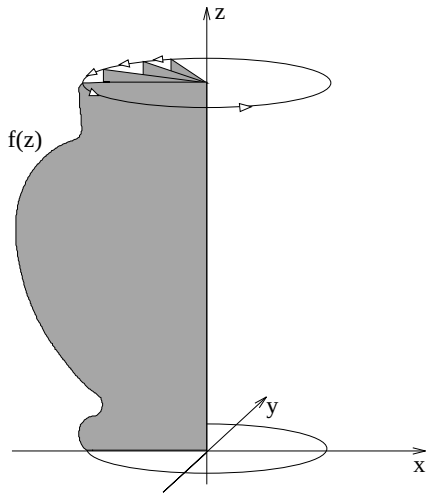


Figure 1.1: *The generating curve $f(z)$.*

1.1 The Object Class of Interest

A surface of revolution results from rotating a function $f(z)$ — the so called *generating curve* — around the z-axis (see figure 1.1) where the z-axis is the axis of symmetry.

In practice, a surface of revolution is the surface of a solid, or object, of revolution. Objects of revolution have been known to man for thousands of years, since the invention of the potter's wheel,[1] and later the lathe, allowed man to produce highly symmetric objects with ease, and today we are surrounded by rotationally symmetric objects such as bottles, pens, vases, glasses, lamp-shades and light-bulbs. Some of the objects of revolution found in every household are shown in figure 1.2.

1.2 The Task

The only image feature used throughout this thesis is a surface's *outline* (also called *apparent contour* [6], *occluding contour*, *profile*, *silhouette* or *limb*), which is the projection of the locus of points on the surface separating the surface's visible image from the occluded parts [6].

The outline is obtained from a grey scale raster image (see figure 1.3a) by

[1]The potter's wheel was invented before 3000 BC in either Sumer or Iran, from where it reached Greece at around 1800 BC, Italy at 750 BC, the upper Rhine basin at 400 BC, Southern England at 50 BC and finally Scotland at 400 AD [23].

Figure 1.2: *Some of the objects of revolution we encounter each day (as found in the office).*

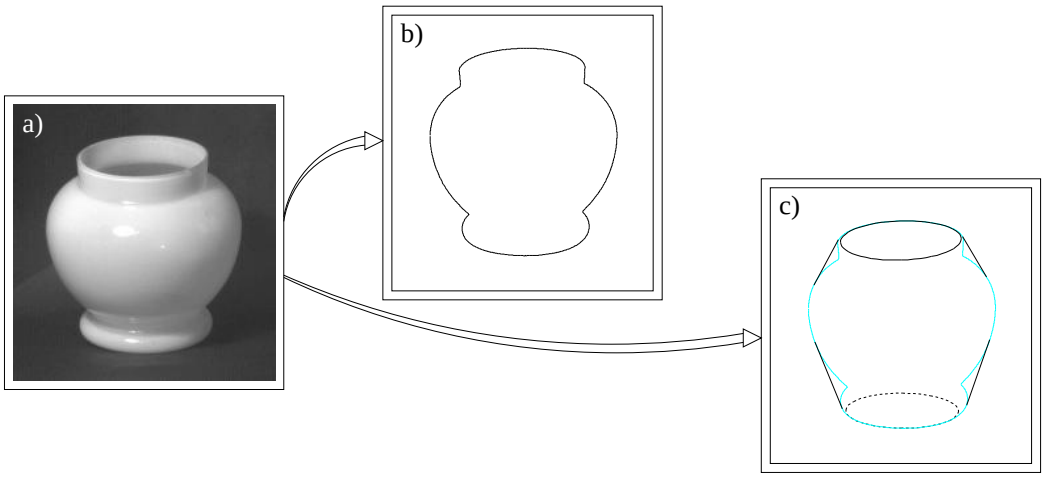


Figure 1.3: *The outline and bitangents are generated from a grey scale raster image.*

a) The grey scale raster image.

b) The surface's outline as found from the raster image.

c) Bitangents to the surface and conics as found from the raster image (in black, the surface's outline is displayed in grey for easy reference).

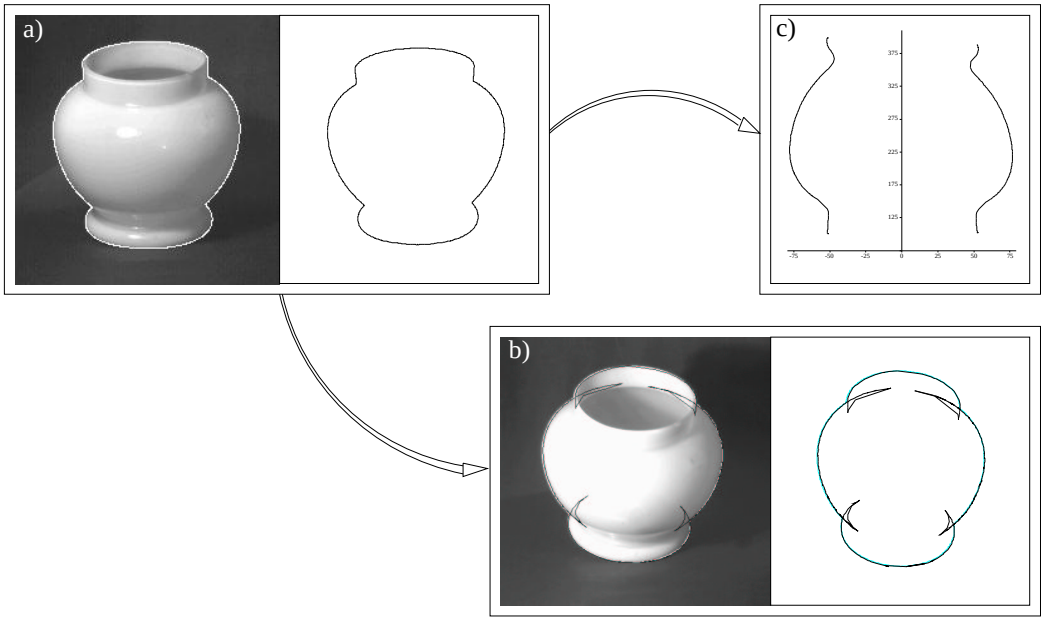


Figure 1.4: *Transfer from image a) onto a second image b), and into a canonical frame c) where invariants can be measured.*
Figures a) and b) each show both an original grey scale image and the outline calculated from it. Figure b) shows also the transferred outline (black). The outlines are nearly identical (the transformation used is described in section 3.2).

applying the simplified Canny [5] edge detector with full hysteresis, providing both step intensity information and edge orientation [17].

Most of this thesis is concerned with:

1. The *transfer* from a surface's outline in one image (figure 1.4a) onto the same surface's outline in any other given view (figure 1.4b) after specifying a minimal number of points in the target image. The transfer is not a simple plane-to-plane transformation, but is rather more complicated (cusps can be created — compare figure 1.4.a to 1.4.b). It should also not be mistaken for the rendering of a known object as viewed under a known viewing direction, as is common in computer graphics application; in general, both the viewing directions in figures 1.4.a) and b), and the surface's generating function, will be unknown. Nonetheless, only very few outline points or features in the target image, such as the top and bottom conics, are used for calculating the transfer. The features used by the various methods are listed in table 1.1 (see also 1.5). The transfer can be used for verification.

Table 1.1: *The features needed by the various methods for calculating the transfer*

	type	bitangent pairs	conics
section 3.2	weak persp.	≥ 2	
section 3.3	weak persp.	≥ 2	
section 3.5	affine	≥ 3	
section 4	affine	≥ 1	1
section 5	projective	≥ 1	2

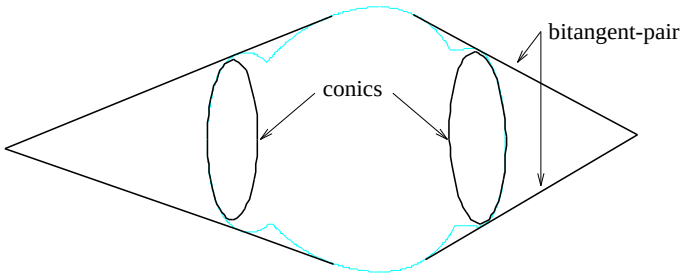


Figure 1.5: *The maximum number of features used for the transfer. 2 bitangent pairs and two conics, both shown in black, are the maximum number of features needed for any of the methods of transfer described in this thesis except section 3.5 (affine extension of scaled orthographic transfer). The outline itself is shown in grey/dotted.*

2. The *transfer* of a surface's outline in one image (figure 1.4a) into a so-called *canonical frame* (figure 1.4c). This allows the retrieval of all or part of the generating function up to a linear transformation from which it is then easy to calculate invariants.

1.3 The Chosen Imaging Geometry

Three different geometries are described in this thesis. Chapter 3 deals with images taken with what is known as the *weak perspective camera*, the approximation of a calibrated camera at infinity by a calibrated camera where the camera–surface distance is much greater[2] than the depth of the surface of revolution.

[2] "Much greater" means approx. 25 times for most of the examples shown throughout this thesis.

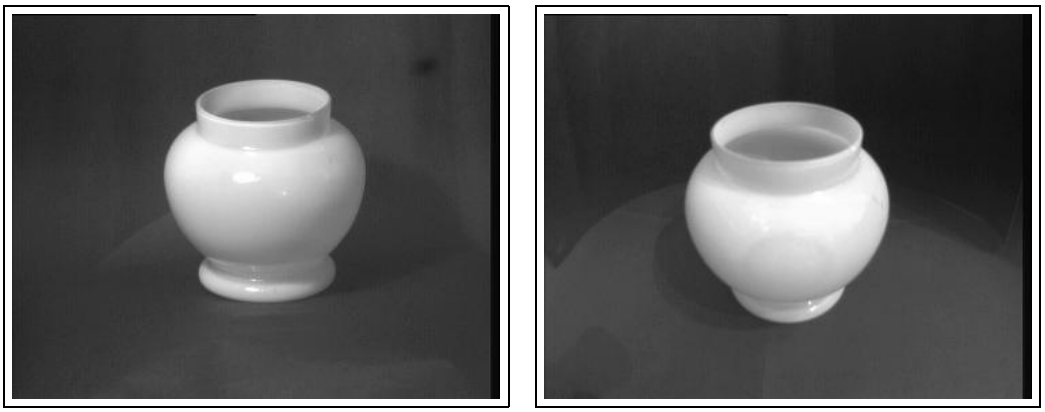


Figure 1.6: *A weak perspective (left) and a projective image of a vase. Note the perspective reduction of the vase's base in the right image.*

Chapter 4, deals with the affine camera, an uncalibrated[3] camera at infinity[4] whose image can than be subject to any affine transformation.[5] Using the model of an affine camera — although without any counterpart in the real world — has the advantages resulting from using a group operation (the plane affine transformations). However, images have still to be taken from "infinity".

The last of these chapters, chapter 5, uses the projective camera, allowing for full perspectivity as well as an uncalibrated camera. This is equivalent to taking an image of the surface with an arbitrarily placed camera (in general not at infinity) and than taking a second image of that image with a camera which is again in an arbitrary position. This is obviously the most general case and accounts for most practical situations.

Examples of both a weak perspective (left) and a projective (fully perspective, right) image of the same surface of revolution are shown in figure 1.6.

1.4 Contributions of this Thesis

Previous work on surfaces of revolution has only exploited isolated points on the outline (such as bitangent points), and has not addressed transfer or verification [12]; or has attempted transfer for calibrated cameras only, using

[3]Uncalibrated only includes linear (affine) distortions, and not such nonlinear distortions as spherical aberration where lines do not project to lines.

[4]Infinity, again, means approx. 25 times the surface of revolution's depth — that is 5 meters!

[5]This corresponds to taking an image of the image with another (uncalibrated) camera at infinity.

a surface's CAD-model [7].

This thesis describes a number of novel methods for transferring the entire outline of a surface of revolution, thereby allowing easy verification as well as the extraction of further invariants from a canonical frame without necessarily requiring a calibrated camera.

The fact that only some isolated points on the outline are needed to calculate the transfer means the methods are suitable for partly occluded surfaces.

1.5 Outline of this Thesis

The following text is divided into six major parts. All the work described in this thesis is based on outlines and makes intensive use of so-called *distinguished points* (as e.g. bitangent points). Chapter 2 gives a brief introduction to the underlying geometry. Most of it is based on [12] (see also [17]). However, some familiarity with homogeneous coordinates as well as projective geometry is assumed. Although an introduction to these subjects can be found in most undergraduate textbooks about computer vision (e.g. [2][6]), or standard literature about projective geometry [22, 24] the best choice is probably the very good appendix of [16].

The next three chapters each describe different methods of transfer for one particular imaging geometry: weak perspective camera (or scaled orthographic projection) in chapter 3, an affine camera in chapter 4 and a projective camera in chapter 5. Each of these parts starts with a section explaining the theoretical background, followed by a short summary, an explanation of the actual implementation and a short section showing some of the results.

Finally chapter 6, discusses how a recognition system could be build using the methods introduced in the previous chapters and giving possible directions for future work.

[6]Be aware that they apply transformations from the right hand rather than the left hand side.

Chapter 2

Distinguished features

Always to be best and distinguished above others.

Homer, *Iliad*, 6

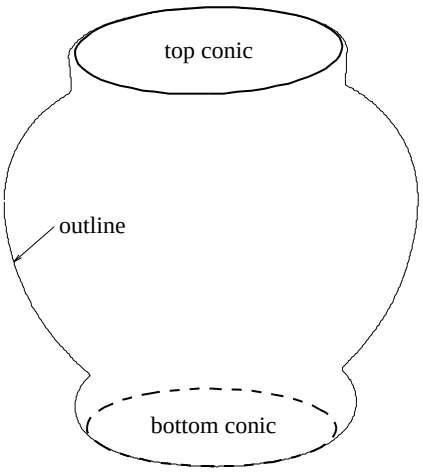


Figure 2.1: *An outline with ending conics.*

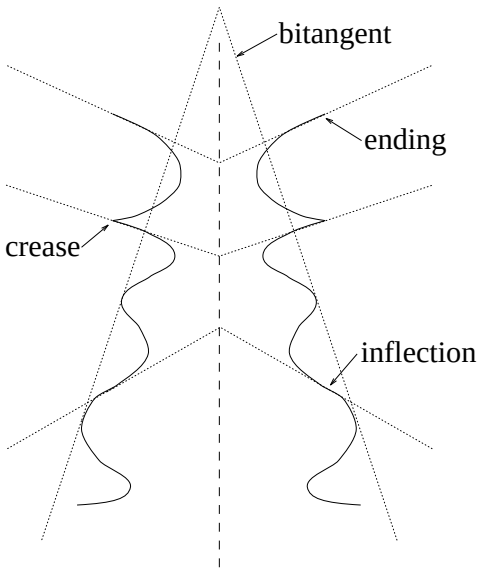


Figure 2.2: *Some distinguished points and the tangents through these points.*

The work described in this paper uses only an outline's *distinguished features*, that is features of the outline that are viewpoint independent. These come in two different categories. The first one are conics, and here mostly an object's top and bottom conic[1] (see figure 2.1). The conics' advantage is that they are comparatively easy identified. The conics' midpoint will be a viewpoint independent point on the axis of symmetry for both the weak perspective and the affine case, however, the concept of a midpoint is meaningless in the projective case.

The second feature are so called (*distinguished points*) on the outline, whose special relationship to a circle on the surface allows their identification both in the image (where they are the projection of points on the circle) and on the surface. Characteristic for all these distinguished points is that they are distinguished by the way their tangent behaves (see figure 2.2). It is therefore necessary to study the tangents first.

[1]Top and bottom conics are not a generic feature for a surface of revolution. However, most man made objects will end abruptly on at least the bottom side, generating a conic when viewed from any other direction than fronto-parallel.

2.1 Tangents

One of the key-properties used throughout this paper is the fact that a tangent to the surface as well as the outline will always intersect the axis of symmetry in exactly the same point, no matter from where the surface is viewed.[2] This is explained below and closely modelled after [12].

2.1.1 The Tangent Cone

For rotationally symmetric surfaces is it possible to formulate one-parameter systems of planes tangent to a circle along the surface. The envelope of these tangent-planes is a right circular cone[3] (see figure 2.3).

The most important result this construction yields is that the apex of every tangent cone lies on the axis of symmetry and that the intersection of a plane tangent to a point on this circle with the axis of symmetry is therefore viewpoint independent. This can be thought of as a map where each circle on the surface is mapped to exactly one point on the axis of symmetry. However, there is no unambiguous map from the axis to the surface, each point on the axis might map onto no, one or many surface points.

The question is what happens to the surface and the tangent cone when projected into an image?

2.1.2 The Outline

The *outline* of a surface in a general perspective projection is a curve in the image given by the set of rays through the camera focal point that are tangent to the surface. The points of tangency on the surface form a space curve — the *contour generator*; see figure 2.4.

An alternative definition of the contour generator is that the plane tangent to the surface at this point passes through the focal point. A result is the following lemma [12]:

Lemma: Except where the image outline cusps,[4] a plane tangent

[2] As long as the viewpoint is outside the surface and the point on the outline is not self-occluded.

[3] The other possibility, a cylinder with circular cross section, is a cone with its apex at infinity; projective geometry doesn't differentiate between points at infinity and more accessible points.

[4] Cusps are ignored in what follows.

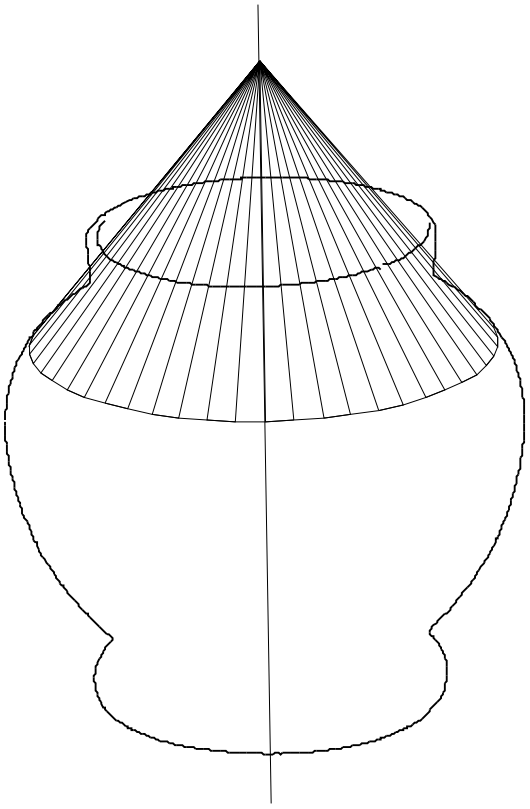


Figure 2.3: *The envelope of all the planes tangent to the points along a circle on the surface is a cone.*

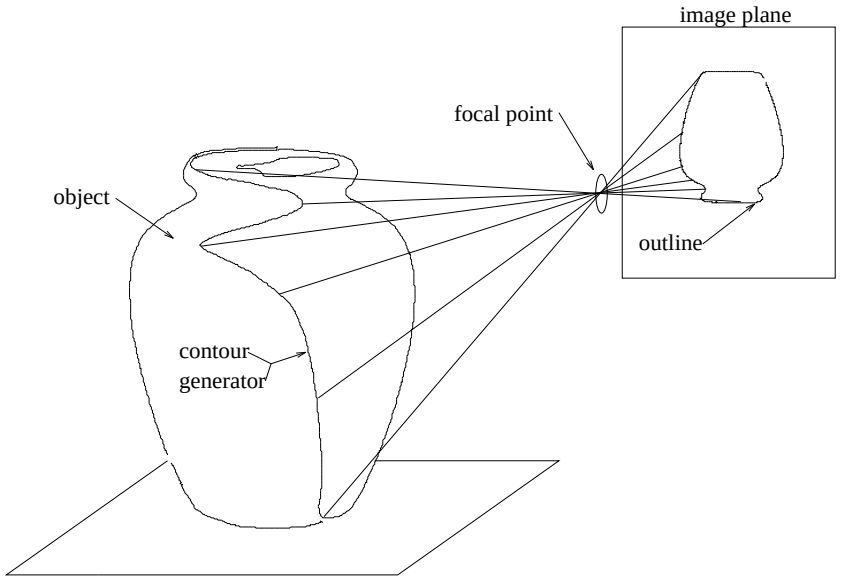


Figure 2.4: *A general projection.*

> to the surface at a point on the contour generator (by definition, such a plane passes through the focal point) projects to a line tangent to the surface outline, and conversely, a line tangent to the outline is the image of a plane tangent to the surface at the corresponding point on the contour generator.

This means that the same properties derived for the surface (viewpoint-invariance of intersections between tangent-planes and the axis of symmetry, map from the surface to the axis) do also exist for the outline, in particular the map from one outline point to one point on the axis of symmetry.

However, it is not easy to see which circle on the surface corresponds to which point on the outline. Only for some *distinguished points* is it possible to establish such a correspondence from the outline alone. Such a relationship exists for example between (see also figure 2.2):

surface	image	confer
bitangent circle	bitangent points	[12, 17]
parabolic circle	inflections	[12, 15]
creases	creases	[12]
endpoints	endpoints	[12]

Of these distinguished points/circles, only (external) bitangent points are used here. This is due to the fact that inflections are hard to find and tend to be subjected to self occlusion rather early, and creases and endpoints, although easy to find, will normally occlude their immediate neighbourhood even under very small viewing angles (thereby making it impossible to find the tangent orientation). Bitangent points as distinguished features are discussed below in more detail.

For bitangents goes [12]:

> **Corollary 1:** A line tangent to the outline at two distinct points is the image of a plane which passes through the focal point, and is tangent to the surface at two distinct points lying on the contour generator.

and

> **Corollary 2:** The intersection of two lines, bitangent to the outline, is a point which is the image of the intersection of the two bitangent planes represented by the lines.

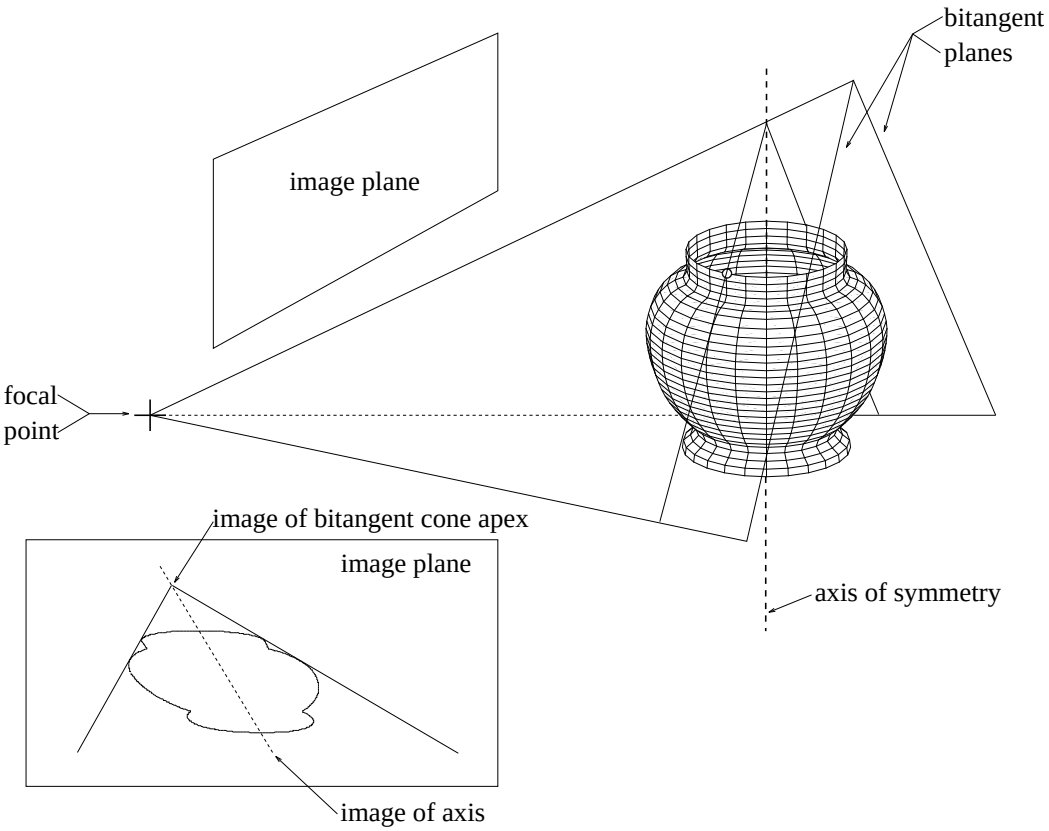


Figure 2.5: *A rotationally symmetric surface, and the planes bitangent to the surface and passing through the focal point.*
It is clear from the figure that the intersection of these planes is a line, also passing through the focal point. Each plane appears as a line in the image. Note that the outline in the image has no symmetry*; This is the generic case [12].*

Figure 2.5 shows this for bitangents tangent to a rotationally symmetric surface.

2.2 The Affine Basis

It has been shown that the position of *bitangent-intersections* on the axis of symmetry — and for the weak perspective or affine case also conic midpoints — are viewpoint independent. Given 2 (3) of these points it is possible to define every point on the axis of symmetry by its position relative to the 2 (affine) or 3 (projective) of these points. Another formulation is that four of points (on a line) form a projective invariant [12] (3 form an affine invariant).

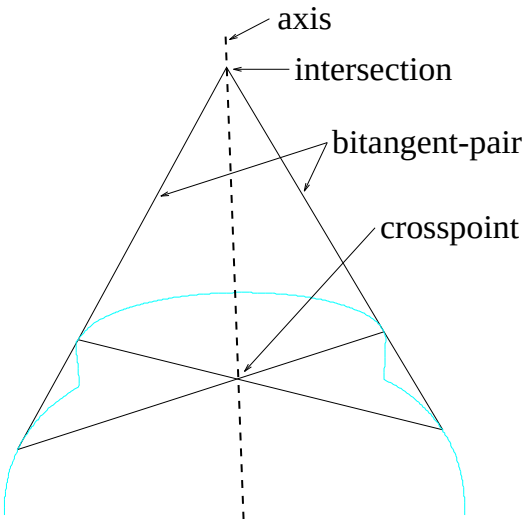


Figure 2.6: *Intersection and crosspoint.*

Chapter 3

The Weak Perspective Camera

Er hat vor Dir gezittert, Tell — Wehe Dir!
Daß Du ihn schwach gesehen, vergibt er nie.

You saw his weakness, and he will never forgive you.

J.C.F. von Schiller, *Wilhelm Tell*, act III sc i

The two methods described in this section allow the transfer from any scaled orthographic projection (this is equivalent to a weak perspective camera) of a surface of revolution onto any other scaled orthographic projection of the same surface, using only 2 bitangent pairs — or any other set of distinguished features that gives one tangent[1] angle with, and two points on, the axis of symmetry.

Both methods are described by first introducing the underlying geometry, especially the dependence of the contour-generator on viewing direction. This is followed by a short discussion of the implementational details and some typical results for each method.

Both methods are then compared to each other and a possible extension to affine projection (angles are not invariant to affine projection) is discussed.

3.1 The underlying Geometry

3.1.1 The Surface of Revolution

Assume a surface's *generating function* $F(Z)$ is known, where the Z-axis is the surface's axis of symmetry in the *object coordinate system*.[2] The equation of the *surface of revolution* $S_F(Z, \Phi)$ that is generated by rotating the generating function around the Z-axis (in homogeneous coordinates) is then

$$S_F(Z, \Phi) = \begin{pmatrix} F(Z)\cos(\Phi) \\ F(Z)\sin(\Phi) \\ Z \\ 1 \end{pmatrix} \tag{3.1}$$

with $0 \leq \Phi < 2\pi$. The family of planes tangent to the surface is

$$N_F(Z, \Phi) = \begin{pmatrix} \cos(\Phi) \\ \sin(\Phi) \\ -F'(Z) \\ F'(Z)\,Z - F(Z) \end{pmatrix} \tag{3.2}$$

(the first 3 coordinates are the *surface normal* in non-homogeneous coordinates). Figure 3.1 shows both the generating function as well as the generated surface of revolution in the object coordinate system.

[1] Tangent to the surface of revolution as well as its outline.

[2] Object coordinates are denoted by capital letters X, Y, and Z.

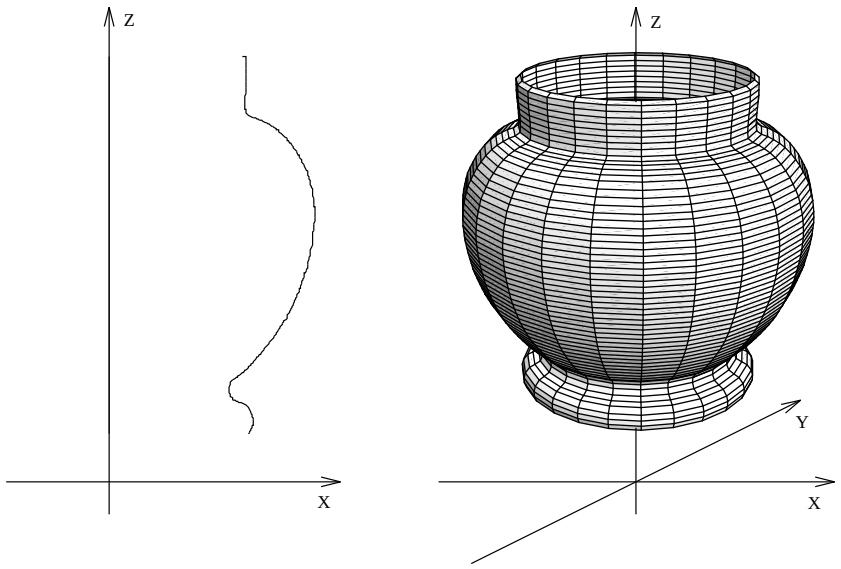


Figure 3.1: *The generating curve $F(Z)$ (left) and the generated surface of revolution $S_F(Z, \Phi)$ (right) in the object coordinate system.*

3.1.2 The Weak Perspective Camera

The weak perspective camera, or scaled orthographic projection, models the process of viewing this surface from a point at infinity.

The *viewpoint* can be expressed as

$$V = \begin{pmatrix} \cos(\Theta)\sin(\Phi) \\ \cos(\Theta)\cos(\Phi) \\ \sin(\Theta) \\ 0 \end{pmatrix} \tag{3.3}$$

with the *azimuth* Φ, the *elevation* Θ and 0 in the last component denoting that the point is at infinity. However, Φ can be chosen arbitrarily since the surface is rotationally symmetric, and $\Phi = \pi$ results in the viewpoint

$$V = \begin{pmatrix} 0 \\ -\cos(\Theta) \\ \sin(\Theta) \\ 0 \end{pmatrix} \tag{3.4}$$

One parameter, the elevation Θ, is therefore sufficient to describe the viewpoint. Another interpretation is that the elevation Θ is the *viewing direction*. It is $0 \leq \Theta < \pi/2$ since the outline as viewed from Θ_1 is exactly the same as viewed from $\Theta_2 = -\Theta_1$[3] and the outline viewed at $\Theta = \pi/2$ is only a circle.

[3]This is true for both scaled orthographic and affine, though not perspective, projection.

The surface's *contour generator* $\Gamma_{F,\Theta}(Z)$ is formed by the points of tangency between the surface and lines in the viewing direction (the plane $N_F(Z,\Phi)$ tangent to the surface has to pass through the viewpoint V). These are all the points with

$$\begin{aligned} V^T N_F(Z,\Phi) &= 0 \\ -\sin(\Phi)\cos(\Theta) - F'(Z)\sin(\Theta) &= 0 \\ \sin(\Phi) &= -F'(Z)\tan(\Theta) \qquad (3.5) \\ \Longrightarrow \quad \cos(\Phi) &= \pm\sqrt{1 - F'(Z)^2\tan^2(\Theta)} \qquad (3.6) \end{aligned}$$

where the use of the "$\pm$" sign indicates the fact that the contour generator when viewed from a viewing direction with $X = 0$ is symmetric with respect to the YZ-plane.

Substituting equations 3.5 and 3.6 into equation 3.1 gives the equation for the contour generator

$$\Gamma_{F,\Theta}(Z) = \begin{pmatrix} \pm F(Z)\sqrt{1 - F'(Z)^2\tan^2(\Theta)} \\ -F(Z)F'(Z)\tan(\Theta) \\ Z \\ 1 \end{pmatrix} \qquad (3.7)$$

The contour generator is a space curve which, for generic generating functions, will be smooth for all except a few points but not continuous (since it isn't even defined for $1 - F'(Z)^2\tan^2(\Theta) < 0$).

Once the contour generator is given it is then possible to calculate the *outline* $\gamma_{F,\Theta}(y)$ — the projection of the contour generator onto a plane *perpendicular* to the viewing direction. Choosing the plane through the origin such that the x-axis of the *image coordinate system*[4] coincides with the X-axis in the object coordinate system, and the y-axis is the Z-axis' image (see figure 3.2) is equivalent to a multiplication with the matrix of projection

$$\mathtt{P} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad (3.8)$$

This results in the outline being

$$\gamma_{F,\Theta}(Z) = \begin{pmatrix} \pm F(Z)\sqrt{1 - \tan^2(\Theta)F'(Z)^2} \\ \cos(\Theta)\left(Z - F(Z)F'(Z)\tan^2(\Theta)\right) \\ 1 \end{pmatrix} \qquad (3.9)$$

which, of course, is also a non-continuous, non smooth function. It is indeed characteristic for this function to form cusps as can be seen, for example, in figure 3.2.

[4]Image coordinates are denoted by small letters x and y.

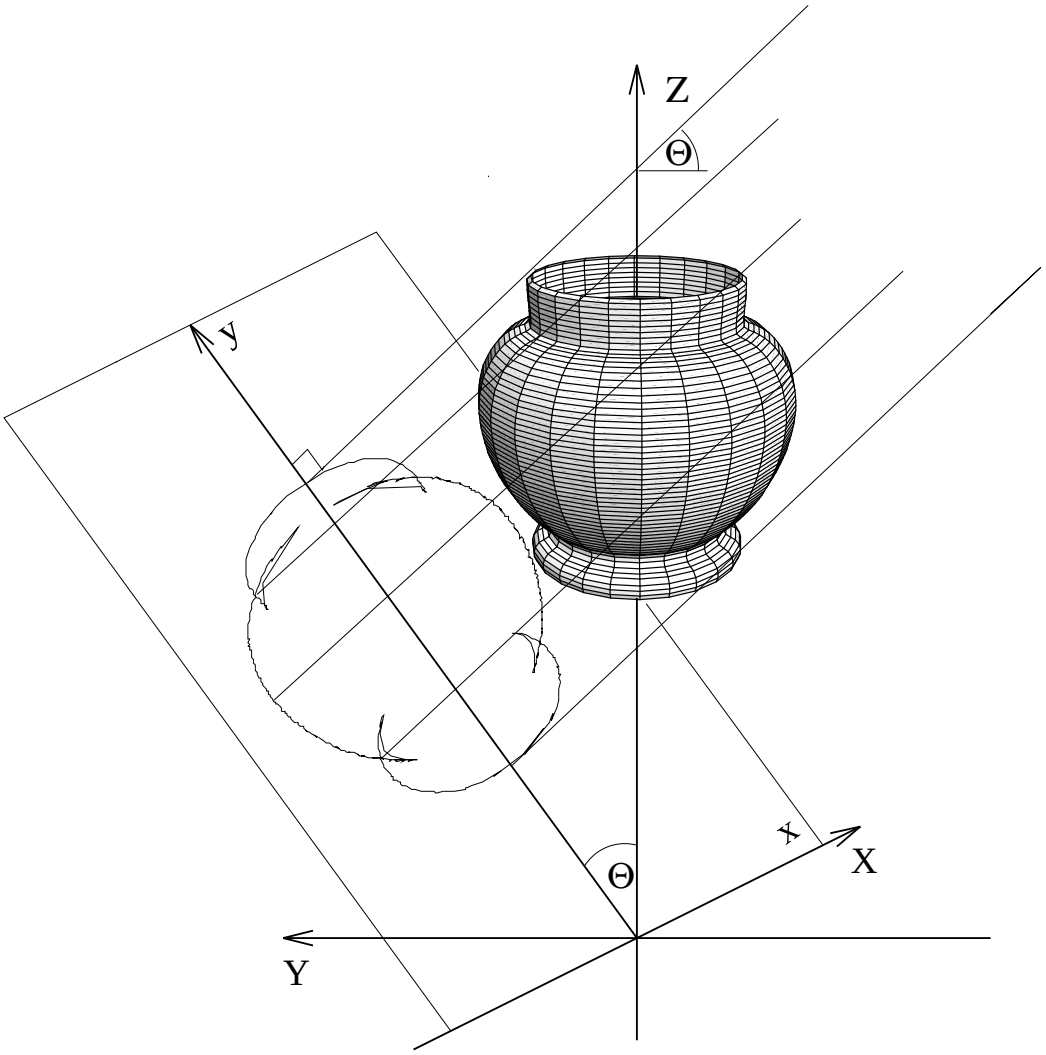


Figure 3.2: *Orthographic projection.*
The projection onto a plane perpendicular to the viewing direction Θ *(indicated by rays).*
The figure also shows the connection between the object coordinate system (X, Y, Z) *and the image coordinate system* (x, y)*.*

It is clear that for $\Theta = 0$, i.e. a viewing direction perpendicular to the axis of symmetry (*fronto-parallel*), the outline is

$$\gamma_{F,0}(Z) = \begin{pmatrix} \pm F(Z) \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} \pm f(u) \\ u \\ 1 \end{pmatrix} = \gamma_{f,0}(u) \tag{3.10}$$

so this here is a way to retrieve the generating function, where u is some parameterisation.

Equation 3.8 is of course only one special orthographic projection. In general both the object and image coordinate systems will not share the same origin (that is: they are translated against each other) and will also be rotated against each other. Also the studied effect is scaled orthographic projection, so an additional scale-factor is needed. All these effects are equivalent to applying a similarity transform after projection; it is a transformation of the form

$$\mathtt{T}_{\text{sim}} = \begin{pmatrix} s\,\cos(\beta) & s\,\sin(\beta) & t_x \\ -s\,\sin(\beta) & s\,\cos(\beta) & t_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.11}$$

where β is a rotational angle, s a scale factor, and (t_x, t_y) a translation.

The final equation for the outline as observed in an image will therefore be

$$\tilde{\gamma}_{F,\Theta}(Z) = \mathtt{T}_{\text{sim}}\,\gamma_{F,\Theta}(Z) \tag{3.12}$$

3.1.3 Recovering the Generating Function

It has been shown in section 3.1.2 that given the generating function $F(Z)$, the viewing direction Θ, and the similarity transformation $\mathtt{T}_{\text{sim}}$, it is possible to generate any scaled orthographic view of the surface $\tilde{\gamma}_{f,\Theta}(u)$ (equation 3.12).

It is therefore not very surprising that it is possible to reverse this process and retrieve the generating function $F(Z)$ (the part which is not self-occluded) when given $\tilde{\gamma}_{f,\Theta}(u)$, $\mathtt{T}_{\text{sim}}$, and Θ. But even if the transformation $\mathtt{T}_{\text{sim}}$ is not given is it still possible to calculate at least a scaled version $f(u)$ of the generating function $F(Z)$ with

$$f(u) = s_1\,F(Z) \tag{3.13}$$

where s_1 is an unknown scaling factor (not necessarily the same as in $\mathtt{T}_{\text{sim}}$), since it is always possible to undo both the rotation and translation if the locations of two points on the axis of symmetry are known. This is done

by calculating the similarity transform that maps these two points onto two arbitrarily chosen points on the axis of symmetry (this will be called the normalised frame). It will later be shown how the scaling can be taken care of by mapping a two views onto each other.

After normalising, the outline's equation is then

$$\gamma_{f,\Theta}(u) = \begin{pmatrix} \pm f(u)\sqrt{1-\tan^2(\Theta)\,f'(u)^2} \\ \cos(\Theta)\,(u - f(u)\,f'(u)\tan^2(\Theta)) \\ 1 \end{pmatrix} = \begin{pmatrix} x_o(u) \\ y_o(u) \\ 1 \end{pmatrix}. \tag{3.14}$$

where the subscript 'o' stands for outline.

The outline's tangent orientation $f'_o(u) = \frac{\partial x_o(u)}{\partial y_o(u)}$ is related to the generating function's tangent orientation $f'(u)$ by the formula[5]

$$f'_o(u) = \frac{f'(u)}{\sqrt{\cos^2(\Theta) - \sin^2(\Theta) f'(u)^2}} \tag{3.15}$$

Given the outline $(x_o(u), y_o(u), 1)$, and the viewing direction Θ, and provided the outline's tangent orientation can be calculated in every point, it is possible to retrieve the generating function (except for self occlusion, especially where the outline cusps) by first solving equation 3.15 for $f'(u)$:

$$f'(u) = \frac{f'_o(u)\,\cos(\Theta)}{\sqrt{1 + \sin^2(\Theta)\,f'_o(u)^2}} \tag{3.16}$$

with $0 \leq \Theta < \pi/2$, and afterwards solving the equation for $x_o(u)$ (equation 3.14) for the generating function $f(u)$:

$$f(u) = \frac{x_o(u)}{\sqrt{1 - \tan^2(\Theta)\,f'(u)^2}} \tag{3.17}$$

The parameterisation u can finally be recovered from the equation for $y_o(u)$ in equation 3.14; it is

$$u = \frac{y_o(u)}{\cos(\Theta)} + f(u)\,f'(u)\,\tan^2(\Theta) \tag{3.18}$$

It is clear that $f(u)$ has to be a scaled and along the axis of symmetry translated version of $F(Z) = \frac{1}{s_1} f(u)$ and $u = Z = s_1 \cdot y + t_y$ since the only transformation applied to the outline was a similarity transformation and since both the rotation and translation in x-direction had been undone.

[5]No term f'' is involved (as can easily be proved). That they indeed have to cancel each other out can be seen from the fact that it is possible to derive the equation from transferring each tangent cone individually — and a tangent cone's generating function's second order derivative is zero.

3.1.4 How to calculate the viewing direction

So far the viewing direction was assumed to be known. However, the viewing direction will be unknown in almost all practical situations and therefore has to be recovered first. Using only two points on, and one angle with, the axis of symmetry, this is not possible from a single view. However, given a fronto-parallel *reference view* of the surface, it is possible to find the same 2 distinguished points on, and the tangent angle α with, the axis of symmetry in both views (see figure 3.3). The equation analogue to equation 3.15 is then

$$\tan(\alpha_o) = \frac{\tan(\alpha)}{\sqrt{\cos^2(\Theta) - \sin^2(\Theta)\tan^2(\alpha)}} \tag{3.19}$$

where α is the angle in the reference view, and α_o the corresponding angle in the view that is to be transferred. This can be solved for the viewing direction $0 \leq \Theta < \pi/2$ with

$$\Theta = \arccos\left(\sqrt{\frac{\tan^2(\alpha) + \tan^2(\alpha)\ \tan^2(\alpha_o)}{\tan^2(\alpha_o) + \tan^2(\alpha_o)\ \tan^2(\alpha)}}\right) \tag{3.20}$$

for $\alpha_o \geq \alpha$ (which will always be the case).

3.1.5 Transfer using two arbitrary views

However, needing a fronto-parallel view of the surface is a severe restriction. It would be much more convenient if it were possible to effect the transfer between any two views without knowing what the viewing directions are.

This is in fact possible by using a *virtual viewing direction*, or *virtual angle* θ:

$$\theta = \arccos\left(\sqrt{\cos^2(\Theta_1)\ \cos^2(\Theta_2)}\right) \tag{3.21}$$

where Θ_1 and Θ_2 are the two viewing directions. The basic idea behind this is:

Given two outlines $(x_{o1}(u), y_{o1}(u), 1)^T$ and $(x_{o2}(u), y_{o2}(u), 1)^T$ belonging to two different viewing directions Θ_1 and Θ_2, equations 3.14 and 3.15 can basically be written as

$$x_{oi}(u) = \pm f(u)\sqrt{1 - tan^2(\Theta_i)\, f'(u)^2} \tag{3.22}$$

$$y_{oi}(u) = \cos(\Theta_i)\left(u - f(u)\, f'(u)\ \tan^2(\Theta_i)\right) \tag{3.23}$$

$$f'_{oi}(u) = \frac{f'(u)}{\sqrt{\cos^2(\Theta_i) - \sin^2(\Theta_i) f'(u)^2}} \tag{3.24}$$

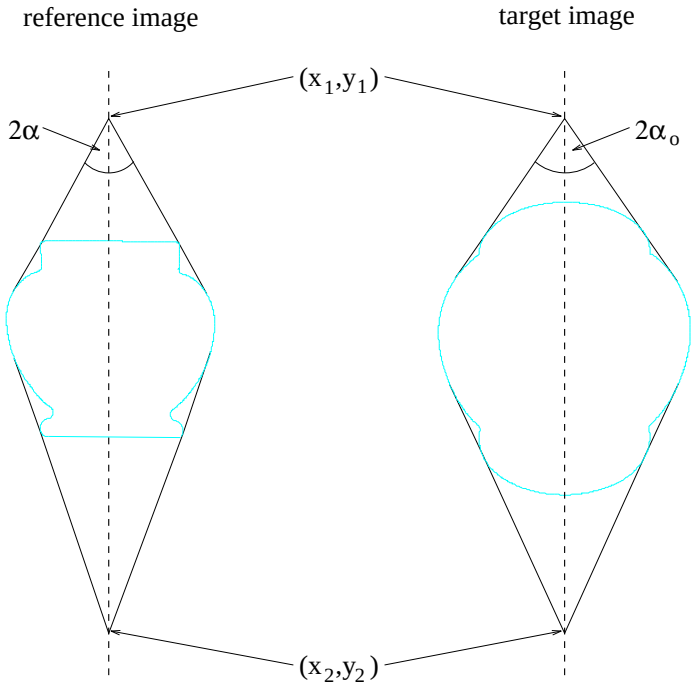


Figure 3.3: *Two points on the axis of symmetry* (x_1, y_1) *and* (x_2, y_2) *as well as one tangent angle* (α *or* α_o) *are needed to calculate the viewing direction. The figure on the left (reference view) is a fronto-parallel view.*

For $\Theta_1 < \Theta_2$[6] it is now possible to pretend that the first view is a fronto-parallel view, that is

$$\tilde{f}(\tilde{u}) = x_{o1}(u) \tag{3.25}$$
$$\tilde{u} = y_{o1}(u) \tag{3.26}$$
$$\tilde{f}'(\tilde{u}) = f'_{o1}(u) \tag{3.27}$$

Substituting $\tilde{f}(\tilde{u})$ for $f(u)$ and $\tilde{u}$ for u in equations 3.22–3.24 leads finally to

$$\begin{aligned} x_{o2}(u) &= \pm x_{o1}(u)\sqrt{1-\tan^2(\Theta_2)\, f'_{o1}(u)} \\ &= \pm f(u)\sqrt{1-\frac{1-\cos^2(\Theta_1)\,\cos^2(\Theta_2)}{\cos^2(\Theta_1)\,\cos^2(\Theta_2)} f'(u)} \end{aligned} \tag{3.28}$$

$$\begin{aligned} y_{o2}(u) &= \cos(\Theta_2)\left(u - x_{o1}(u)\, f'_{o1}(u)\, \tan^2(\Theta_2)\right) \\ &= \cos(\Theta_1)\,\cos(\Theta_2)\left(u - f(u)\, f'(u)\, \frac{1-\cos^2(\Theta_1)\,\cos^2(\Theta_2)}{\cos^2(\Theta_1)\,\cos^2(\Theta_2)}\right) \end{aligned} \tag{3.29}$$

$$\begin{aligned} f'_{o2}(u) &= \frac{f'_{o1}(u)}{\sqrt{\cos^2(\Theta_2) - \sin^2(\Theta_2)\, f'_{o1}(u)}} \\ &= \frac{f'(u)}{\sqrt{\cos^2(\Theta_1)\,\cos^2(\Theta_2) - (1-\cos^2(\Theta_1)\,\cos^2(\Theta_2))\, f'(u)^2}} \end{aligned} \tag{3.30}$$

3.2 Method 1. Using the Generating Curve

3.2.1 Summary

It has been shown that, given a scaled orthographic image of a surface of revolution and the viewing direction, it is possible to calculate the surface's generating function. Given two images, the *reference* and the *target view*, neither necessarily fronto-parallel, it is possible to calculate a virtual viewing direction between the two images. This allows the transfer from the reference view onto the target view and vice versa. However, the use of two views is necessary, the generating function can not be retrieved.[7]

All that is needed in each view to transform one onto the other are two points on the axis of symmetry (e.g. bitangent-intersection), one tangent angle with

[6] This is equivalent to assuring $f'_{o1}(u_0) < f'_{o2}(u_0)$. Only in this case is it possible to solve for a virtual angle θ.

[7] This means especially that it is not possible to transfer into a canonical frame and calculate invariants from this representation, as can be done with the methods discussed in section 4 and 5.

the axis of symmetry, and of course, for each point on the outline, the x- and y-coordinates plus the angle between a tangent to the outline at this point and the positive y-axis (axis of symmetry). It is then easy to transfer the outline in one view onto the reference view by

1. Applying equations 3.16–3.18 pointwise.
2. Transforming the transferred outline in such a way that it has the same translation, rotation and scaling as the target image.

The transformation in 2 between the target and the reference view can be found in different ways. Two possibilities used are either to map the transferred view's two reference points on the axis of symmetry (see figure 3.3) onto the equivalent points in the target image, or to map points on the transferred outline directly onto the equivalent points on the target outline, e.g. bitangent-points (assuming that their position in both views is known, it is easy to calculate the new position on the transferred outline). The second method results in a much better fit, which is also robust against small perspective distortions (see section 3.4).

3.2.2 The Implementation

The actual implementation is divided into 4 steps, each implemented in a program on its own. The first 2 steps (and to some extent the third) are concerned with acquisition and are used for both this method and the one described in 3.3.

The first step finds the matrix of transformation that will rotate and translate the outline so that its axis of symmetry will coincide with the y-axis. Next the transformation is carried out.

The third stage finds the virtual angle, or viewing-direction, needed for calculating the transformation; and the last stage carries out the actual transfer.

3.2.2.1 Computing the Axis of Symmetry

To transfer an outline point using equations 3.16–3.18, it is necessary that the outline's axis of symmetry coincide with the y axis in the used coordinate system. Also, each point's x_o- and y_o-coordinates are needed, together with the tangent orientation $f'_o = \tan(\alpha_o)$, where α_o is the angle between the tangent at the point (x_o, y_o) and the axis of symmetry.

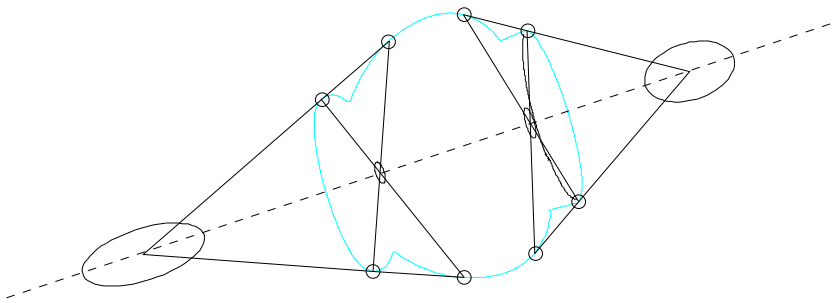


Figure 3.4: *Covariance for intersections and cross-points. The ellipses around the intersections and cross-points denote the maximal error for these intersections, allowing for an error of up to 10 pixels in each tangent-point.*

The continuous edge curves extracted from the Canny edge-detector's output are saved as arrays of n points, each point of the form (x, y, α) where α is the angle between the negative x-axis and the tangent at (x, y). In general, the y-axis will not be the axis of symmetry. It is therefore necessary to rotate and translate the outline that is to be transferred in such a way that the axis of symmetry is identical to the y-axis, and such that the angle α is the angle between the positive y-axis and the tangent.

The axis of symmetry is the regression line through the following points[8]:

- The intersections of bitangent-pairs (at least two bitangent-pairs need to be given; it is possible to use more).
- The cross-points, that is the intersection of two lines drawn through the bitangent-points in such a way that the intersection is inside the surface (see also figure 3.4).

Each intersection or cross-point is calculated from the bitangent-points, and is accompanied by a covariance matrix which is calculated assuming that the bitangent-points have the identity-matrix as covariance matrix. This is shown in figure 3.4, where each of the ellipses around intersections and cross-points represents a covariance matrix.

An iterative algorithm is used to calculate the matrix of the transformation

$$\mathtt{T} = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) & t_x \\ -\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.31}$$

[8] It is actually possible to use a wealth of additional information such as "inter-pair" cross-points or even the whole outline, but it is questionable if the benefits will outweigh the difficulties, especially when dealing with partly occluded outlines.

which minimises the weighted sum of the orthogonal distances of all the intersections and bitangent-points from the y-axis.

Applying the transformation T to a point yields

$$T \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\varphi)\, x + \sin(\varphi)\, y + t_x \\ -\sin(\varphi)\, x + \cos(\varphi)\, y \\ 1 \end{pmatrix} \tag{3.32}$$

where $\cos(\varphi)\, x + \sin(\varphi)\, y + t_x$ is the distance from the y-axis. Applying the same transformation to the point's covariance-matrix leads to a new covariance-matrix

$$\begin{pmatrix} \tilde{\sigma}^2_{xx} & \tilde{\sigma}^2_{xy} \\ \tilde{\sigma}^2_{xy} & \tilde{\sigma}^2_{yy} \end{pmatrix} = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi) & \cos(\varphi) \end{pmatrix} \begin{pmatrix} \sigma^2_{xx} & \sigma^2_{xy} \\ \sigma^2_{xy} & \sigma^2_{yy} \end{pmatrix} \begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix} \tag{3.33}$$

and the upper left element of that matrix,
$\tilde{\sigma}^2_{xx} = \sigma^2_{xxi}\cos^2(\varphi) + 2\sigma^2_{xyi}\cos(\varphi)\sin(\varphi) + \sigma^2_{yyi}\sin^2(\varphi)$ is the point's covariance in the x-direction which is used as a weight.

The function to minimise is therefore

$$\min_{\varphi, t_x} \sum_{i=0}^{n-1} \frac{\left(x_i \cos(\varphi) + y_i \sin(\varphi) + t_x\right)^2}{\sigma^2_{xxi}\cos^2(\varphi) + 2\sigma^2_{xyi}\cos(\varphi)\sin(\varphi) + \sigma^2_{yyi}\sin^2(\varphi)} \tag{3.34}$$

Several other methods for finding the axis of symmetry and consequently a transformation that maps this axis onto the y-axis are conceivable, such as finding the similarity/affine/projective transformation that maps points on one side of the outline onto points on the other side.

However, even very simple methods using no weights at all proved adequate, indicating that this is not a critical step.

Once the matrix of transformation is found it is than straightforward to implement the actual transformation

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{\alpha} \end{pmatrix} = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) & 0 \\ -\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ \alpha \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ \varphi \end{pmatrix}$$

where $\tilde{\alpha}$ is calculated in degrees (denoted by $^\circ$). Figure 3.5 shows the result for two different surfaces.

3.2.2.2 The Direction of View

The virtual angle θ between the two views is found by solving equation 3.19. The implementation makes use of the four or more angles, provided by the

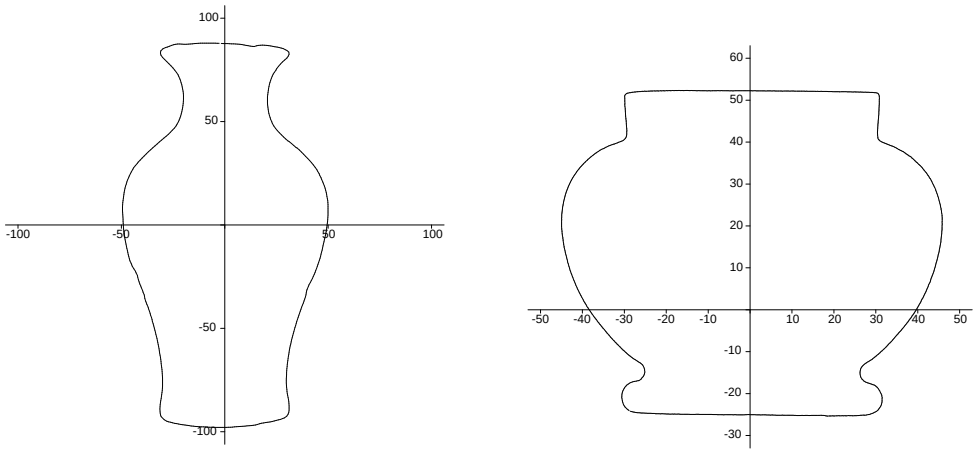


Figure 3.5: *Two reference contours in normalised position*

two or more bitangent-pairs, by solving multiple equations

$$(1 + \tan^2(\alpha_{i1}))\cos^2(\theta) = \frac{\tan^2(\alpha_{i1})}{\tan^2(\alpha_{i2})} + \tan^2(\alpha_{i1}) \tag{3.35}$$

using the pseudo inverse.[9] A solution for θ can only be found for $\tan^2(\alpha_1) < \tan^2(\alpha_2)$.

The same program also calculates a transformation that relates the transferred outline to the reference outline. It first transfers all bitangent-points using equations 3.16–3.18. It then calculates the *affine* transformation which maps all the transferred bitangent-points onto the bitangent points in the reference view in a least square sense, using a pseudo inverse (see figure 3.6).

An affine transformation is used for two reasons. It is not only easier to compute than a similarity transformation, but allows a much closer map between the two views if they show (small) affine or projective distortions and thereby makes the algorithm very robust against these distortions (compare sections 3.2.3 and 3.3.3).

3.2.2.3 Transfer

Once the outline is in the required form (x_o, y_o, α_o) (the first two steps) and the virtual viewing direction is known (the third step), it is immediately possible to calculate the transferred outline by applying equations 3.16–3.18

[9] In this case this is equivalent to calculating the mean value of $\cos^2(\theta)$ over the four angles.

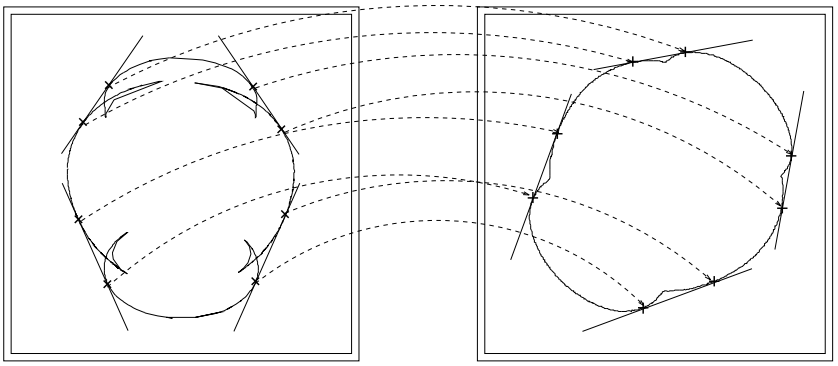


Figure 3.6: *An affine transformation is calculated which maps the transferred bitangent-points onto the reference view's bitangent-points.*

Once the outline is transferred it is transformed using the affine transformation calculated in the third step and can be compared to the reference view. Examples for the transfer from a reference onto the target view are shown in figure 3.7.

3.2.3 Results

Figure 3.7 shows four typical transfers using the method just described. The transferred outline is shown in dotted/grey, while the reference outline is shown in black. The transfer is surprisingly good, given that the angles used (output from the canny edge finder) are not very accurate, and that the images used show considerable perspective distortion (cf section 3.3.3, where this poses a major problem). The method used here did indeed prove to be fairly robust when dealing with faulty data and projective distortions.

Also shown in figure 3.7 are events such as cusps, which denote the start of self-occlusion, and swallowtails (though these are only actually visible on semi-transparent objects).

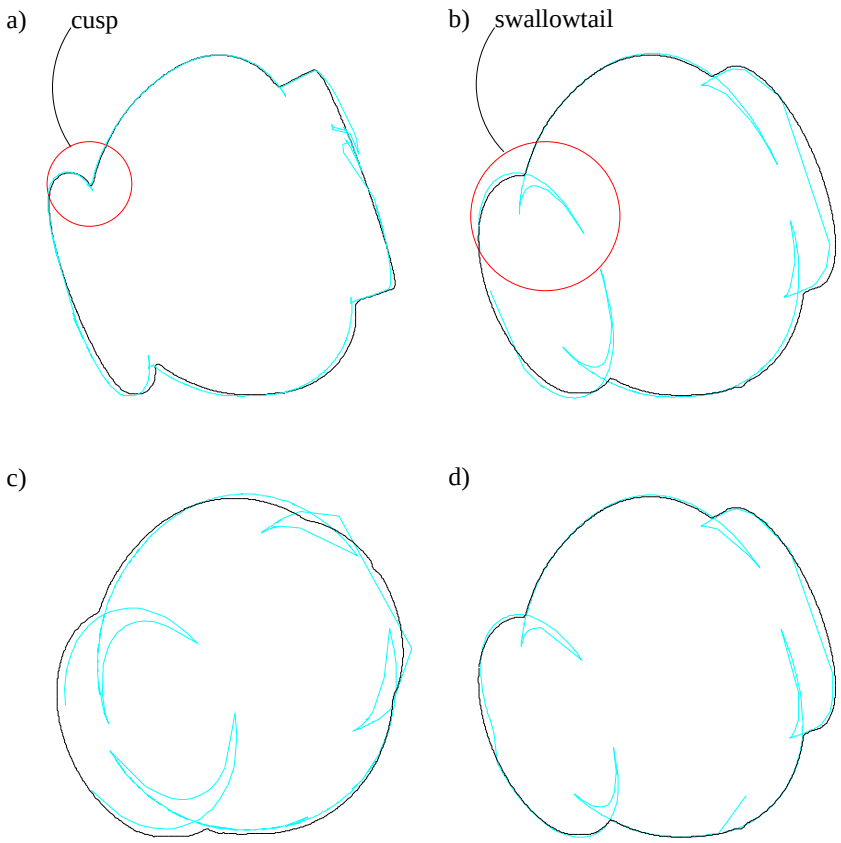


Figure 3.7: *Example of outlines transferred using the generating curve. The images show the transfer of a surface's outline on outlines of the same surface viewed under a larger viewing angle. The transferred outline is displayed in grey/dotted, while the contour it is transferred onto is shown in black. The following table gives the approximate angles:*

	from	*to*
a)	$0°$	$10°$
b)	$0°$	$30°$
c)	$0°$	$-45°$
d)	$10°$	$30°$

.

The development of swallowtails and events such as cusps is typical for the transfer in the direction of increasing angle of view (compare [15]) and denotes self-occlusion.

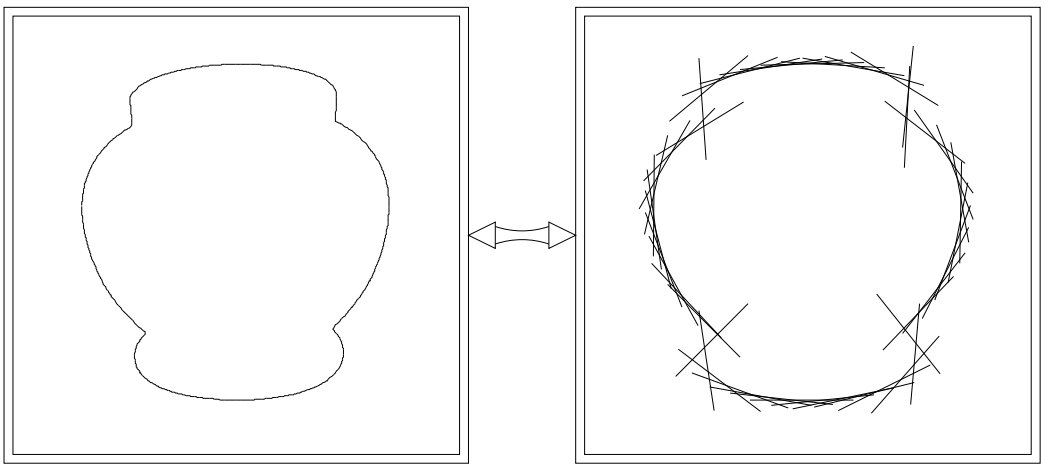


Figure 3.8: *A dual representation for an outline (left) is to save all its tangents (right). The outline is the tangents' envelope.*

3.3 Method 2. Using the Outline's Envelope

It has been shown in section 2 that each line tangent to the outline is the image of a plane tangent to the surface, and especially the contour generator, and that its intersection with the axis of symmetry is viewpoint independent. This fact will be exploited when transferring an outline as described below.

3.3.1 The underlying Geometry

The outline can be seen as the envelope of all its tangents (see figure 3.8). A dual representation for an outline is therefore to save all its tangents rather than the actual points on the outline. A conceivable representation for each tangent would be its point of intersection with, as well as the angle between, the tangent and the axis of symmetry.

3.3.1.1 How to Represent the Intersection

Lengths on a line are not invariant under a *scaled* orthographic projection. However, it has been shown[10] in a number of standard texts (e.g. [14]) that the ratio of length on parallel lines is invariant to scaled orthographic projection.[11] Given two points $(x_1, y_1, 1)$ and $(x_2, y_2, 1)$ on the axis of symmetry, it is therefore possible to save a point on the axis by giving only one coordinate

[10] And is indeed very easy to prove.

[11] It is indeed invariant even under affine projection.

λ, where the point is

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \lambda \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \end{pmatrix} \tag{3.36}$$

(the so-called *affine basis*, indicating that this representation can not only be used for weak perspective but also for affine projection). Determining the transfer of this point is trivial once the position of the two reference points is known in both views. No other relationship between the two views (such as the virtual viewing direction) needs to be known.

3.3.1.2 The Angle

Unlike the ratio of length on the axis of symmetry, the angle between the tangent line and the axis of symmetry is viewpoint dependent. The transfer of the angle can be accomplished in exactly the same way as described in section 3.1.3ff.

3.3.1.3 The Envelope

Finding the outline — the transferred tangents' envelope — although not a problem in theory, might pose considerable difficulty in practice. Whilst in theory an infinite number of tangents can be used, so that the envelope is simply the curve formed by all the intersections between neighbouring tangents, in practice only a limited number of tangents[12] is given. And whilst they are generically equi-spaced and close to each other (distance of less than one pixel) in the outline to be transferred, this will not necessarily be the case for the transferred outline, where intersections between neighbouring tangents are easily 100 times the original distance.[13]

Although several different methods for calculating the tangents' envelope are conceivable[14] it proved to be sufficient for most situations to approximate the outline by the intersections of neighbouring tangents connected with straight lines.

3.3.1.4 Summary

Section 3.2 described how to transfer a surface's outline in one view onto the same surface's outline in another view, the so-called reference view, using the

[12] Between 500 and 1000 in most experiments described here.
[13] This is the case for tangents that were close to cusps in the outline.
[14] Using splines is a method that springs to mind.

outline's algebraic description. A second way is described here, using a dual representation of a (smooth) outline, namely by giving its tangents.[15]

Each tangent line can be described by giving one point (on the axis of symmetry) and the line's orientation (relative to the axis of symmetry). Knowing how 2 points on the line of symmetry transfer is sufficient to determine where every other point on the line is transferred to by using the invariance of the ratio of collinear (parallel) length; the tangent angle can be transferred using the techniques discussed in section 3.1.

The transferred outline is the transferred tangents' envelope. Using the intersection between neighbouring tangents is a sufficiently good approximation for most practical cases.

3.3.2 The Implementation

As in the first method, the implementation is divided into 4 steps, the first 2 of which are identical to those described in section 3.2. The only difference is that now these steps are only part of the acquisition-process, which is completed in the last step.

The third step is similar to section 3.2.2.2 insofar as here too a virtual viewing direction and a matrix of transformation are calculated which relate the transferred view to the reference-view.

The last step is partly concerned with acquisition (changing the outline's representation from points on the outline to tangents to this point) and partly with the transfer.

Below, only the last two steps are detailed.

3.3.2.1 The Direction of View

As in section 3.2.2.2 the viewing direction is found by solving multiple equations 3.35 using the pseudo-inverse.

This is followed by calculating the *similarity* transform that maps the intersections of bitangents in the transferred view's frame onto their opposite numbers in the reference view's frame in a least square sense (if more than two bitangent pairs are used). This is shown in figure 3.9.

[15]Strictly speaking there is no duality between giving *some* outline points and *some* tangents using the discrete data that is available in practice.

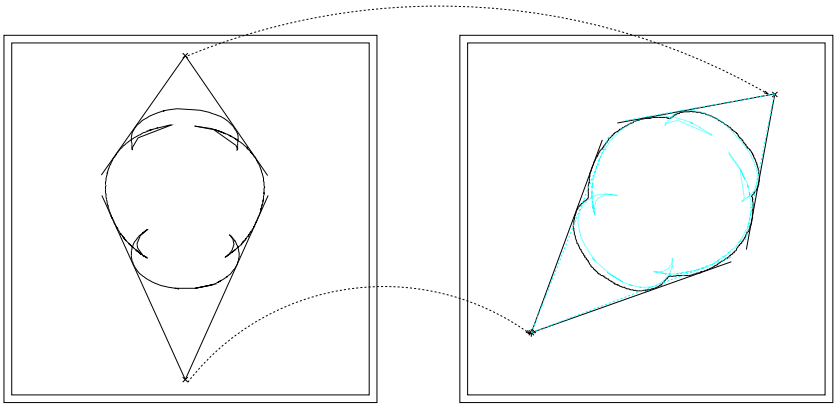


Figure 3.9: *A similarity transformation maps the bitangent-intersections in one image onto their opposite numbers in the second image. Note how an error in the intersections' position (due to perspective distortions) leads to a characteristic offset between the two outlines (the transferred outline is shown dotted).*

This is different from the approach taken in section 3.2.2.2 in more than one respect.

- A similarity rather than an affine transformation is calculated since this method of transfer cannot calculate the position of the bitangent-points after the transfer. Using a similarity transform is in accordance with the assumption that the outlines stem from a scaled orthographic projection. However, it does not take into account the distortions that can be found in real images.
- The intersections of bitangents with the axis of symmetry are in general less accurate than the actual bitangent-points, especially if the tangent's angle with the axis is close to 0 or π (see section 4.4.1).

The two points work together in making the algorithm very sensitive even to extremely small perspective distortions (cf section 3.3.3).

3.3.2.2 Acquisition and Transfer

The acquisition — calculating the intersection between the tangent and the axis of symmetry — is straightforward, and the transfer is done in three steps:

1. A new angle with the axis of symmetry is calculated for each tangent.

2. The intersections between neighbouring tangents are calculated.[16]

3. The intersections are transformed according to the matrix calculated in section 3.3.2.1

3.3.3 Results

Figure 3.10 shows some example transfers, using data similar to those in figure 3.7. The noticeable displacement along the axis of symmetry between transferred and reference contour is not a failure of the algorithm, but is due to the fact that the images used to generate the outlines show some projective distortions for which the algorithm was never developed in the first place. The same is true for figure 3.7 (first method), but here the effect is largely mitigated since points *on the contour* are used to calculate an *affine* transform (as opposed to bitangent intersections far away from the surface determining a similarity transform).

3.3.3.1 The Influence of Perspective Distortions

That the displacement between transferred and target outline in figure 3.10 is indeed due to perspective distortions can be seen when comparing these results to ones achieved with data taken from a distance roughly 25 times the object size (instead of about 7 times) which therefore better approximate the weak perspective camera modelled by the algorithm; figure 3.11 shows this effect.

How a perspective distortion causes the displacement is easily explained by looking at figure 3.12. Tilting the surface by a positive angle causes the top to come slightly closer to the camera while the bottom moves further away. Perspective distortion makes the parts that are coming closer to the camera look slightly bigger, while the parts that moved further away will appear smaller. This is also true for the triangles formed by the two bitangent-pairs: the top one will look bigger, thus the intersection with the axis of symmetry will appear to move away from the surface, while the bottom one will look smaller and the intersection will therefore appear to be closer to the surface. Unfortunately only the intersections get mapped onto each other, and this creates the effect of a displacement along the axis of symmetry.

[16] It is a good idea to disregard intersections between lines with nearly the same orientation as to unreliable, since the accuracy of an intersection between two lines depends on the angle between these lines, and decreases sharply with decreasing angle.

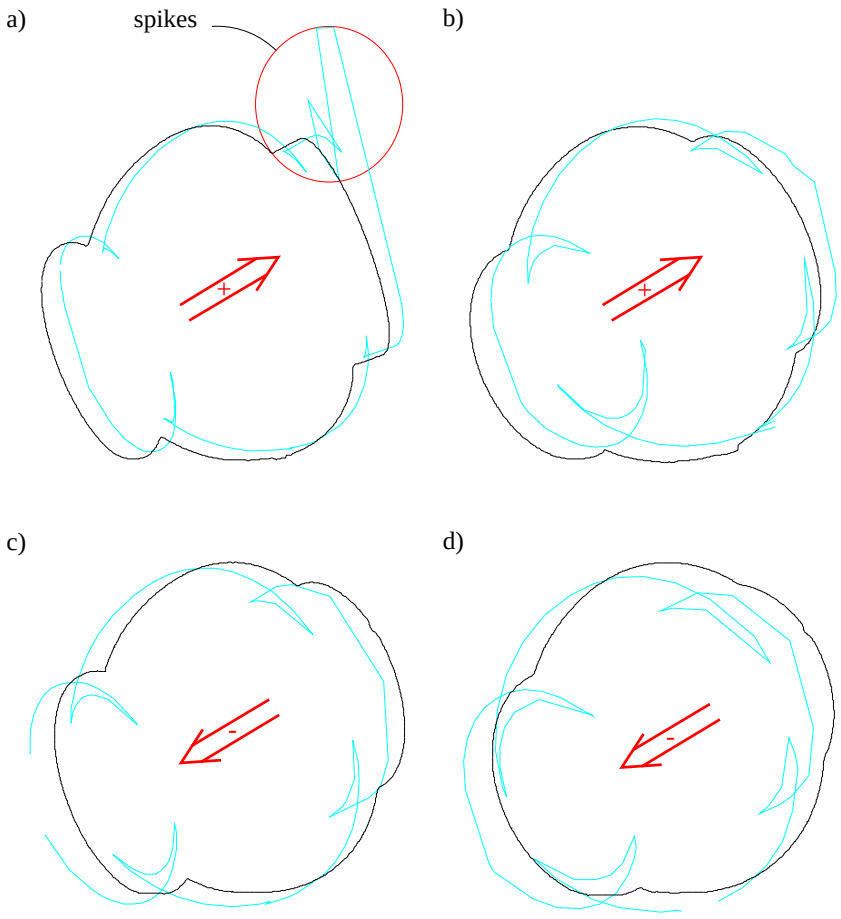


Figure 3.10: *Example of outlines transferred using the envelope*
The images show the transfer of a surface's outline on outlines of the same surface viewed under a larger viewing angle. The transferred outline is displayed in grey/dotted, while the contour it is transferred onto is shown in black. The following table gives the approximate angles:

	from	*to*
a)	$0°$	$20°$
b)	$10°$	$40°$
c)	$0°$	$-30°$
d)	$10°$	$-45°$

Note the spikes in figure a). These are due to errors in the tangent orientations.
Note also the displacement along the axis of symmetry between transferred and reference outline, and how the direction of the displacement depends on the viewing angle's sign. This is due to perspective distortions in the image.

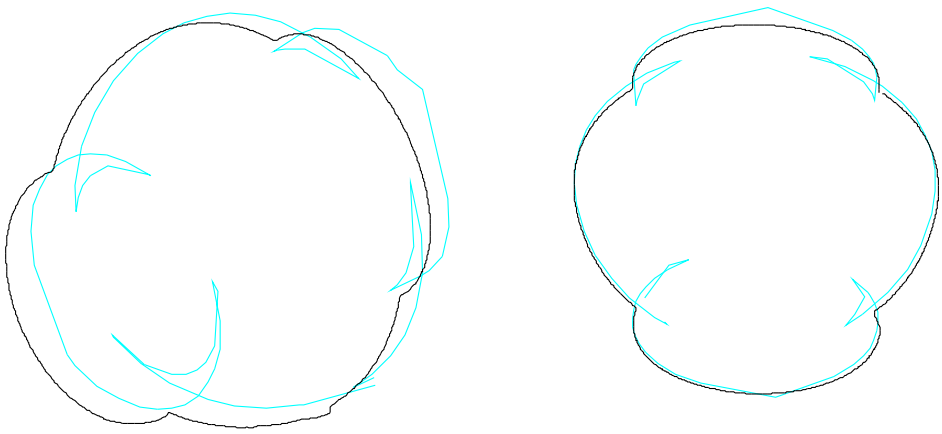


Image taken from a distance approx. 7 times the surface's size

Image taken from a distance approx. 25 times the surface's size

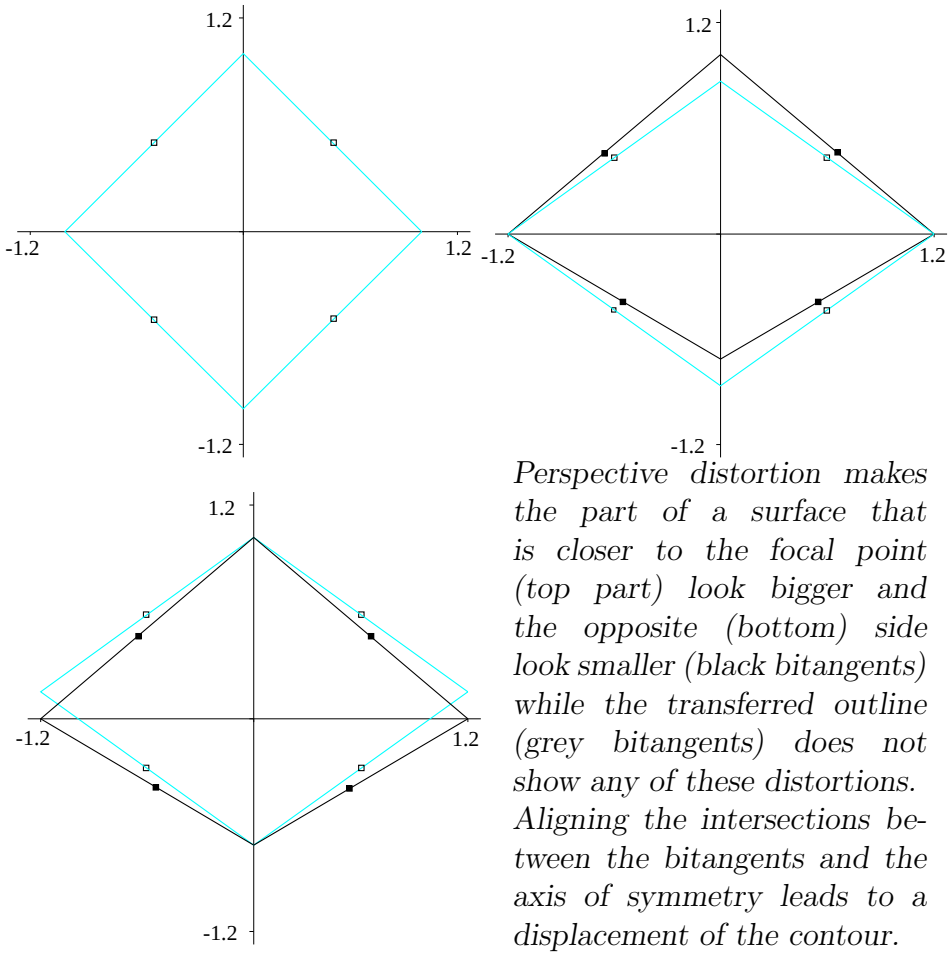
Figure 3.11: *The influence of projective distortions onto the outlines displacement.*
The right outline is belonging to an image taken from 3–4 times the distance the left is taken and therefore better approximating a weak perspective camera. Note the difference in the displacement

3.3.3.2 The Influence of Errors in the Angles

The spikes in figure 3.10 are caused by errors in the tangent angles which, especially if the angles are about equal, can cause the position of the intersection of two neighbouring tangents to be very inaccurate. Two different approaches are possible to eliminate (or at least to reduce) the occurrence of these errors.

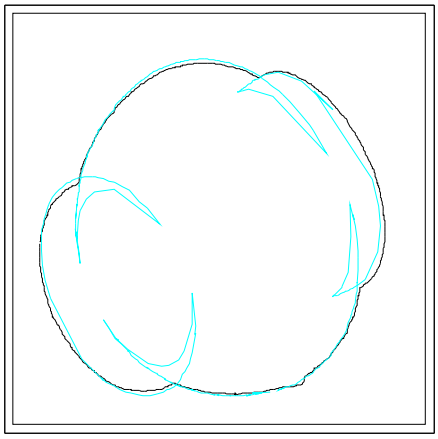
The first possibility is to use better data. The tangent angles used are the ones that are output by a canny edge detector. It is possible to increase the accuracy by interpolating part of the outline around the tangent-point with a quadratic and use the tangent to the quadratic instead. This is indeed exactly how the bitangent-points are found in the first place and is also used in section 4.

The second approach is actually to use an interpolated envelope, rather than the tangent-segments, for example by employing splines. This has the effect of a low pass filter in that it smoothes the edges and greatly reduce spikes.

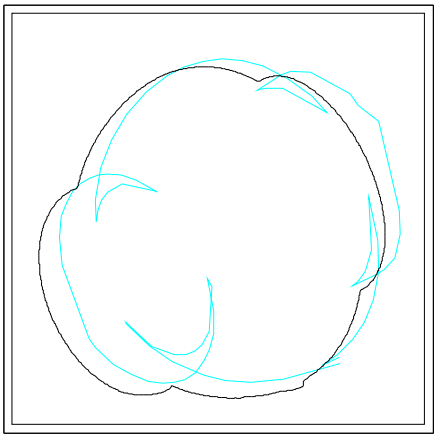


Perspective distortion makes the part of a surface that is closer to the focal point (top part) look bigger and the opposite (bottom) side look smaller (black bitangents) while the transferred outline (grey bitangents) does not show any of these distortions. Aligning the intersections between the bitangents and the axis of symmetry leads to a displacement of the contour.

Figure 3.12: *Displacement caused by projective distortions.*



Transfer using the generating function.



Transfer using the envelope.

Figure 3.13: *Comparing the two methods using the same image data. The images used are taken from a distance roughly 7 times the object size. An approx.* 10°*-view is transferred onto an approx.* 40°*-view.*

3.4 Comparing the two Methods

At first glance — judging from figures 3.7 and 3.10 — it might appear as if the first method (using the generating curve) is clearly superior. It provides not only

1. a tighter fit between transferred and reference outlines (with virtually no displacement) if small perspective distortions are present,
2. but also an apparently greater robustness against errors in the tangent-angles.

To 1 must be added that the first method will nearly always lead to a better fit, even if the images had no perspective distortion at all. This is due to the fact that the intersection between two bitangents is nearly always less accurate than the bitangent-points themselves (see also section 4.4.1). It is, of course, always possible to find bitangents tangent to the transferred outline,[17] though the first method's big advantage is that it provides the position of the transferred bitangent points without requiring additional steps. Figure 3.13 shows the transfer from about 10° to about 40°, both images taken from a distance roughly. 7 times the object size. It is clear that the first method is working much better and more reliable then the second one.

[17] By exactly the same process that found the bitangents tangent to the reference outline.

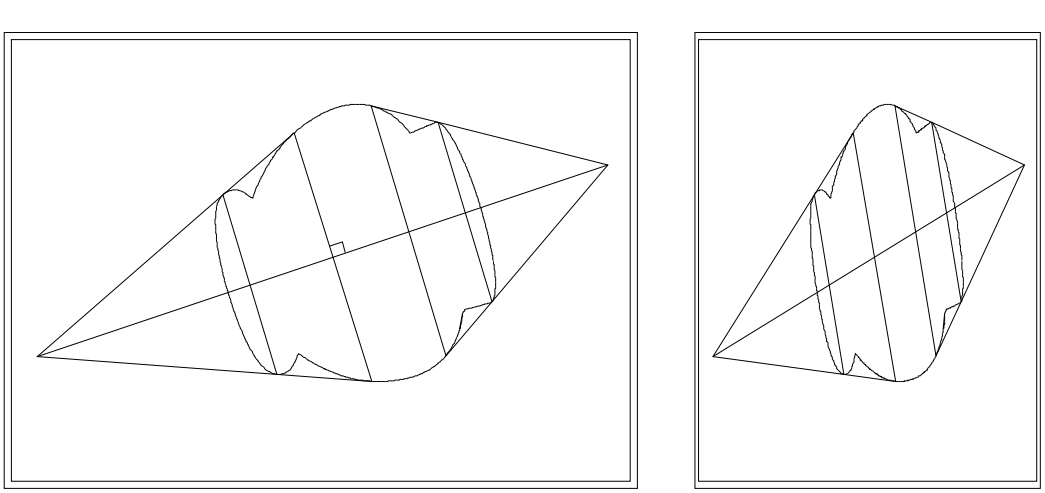


Figure 3.14: *An aspect ratio other than 1 leads to a skewed outline whenever the axis of symmetry is not parallel to the image's x- or y-axis.*

However, the method for the transfer of affinely projected outlines described in section 4 can be seen as a modification of the second method, while the first method can not easily be used for the affine or projective case.

3.5 Affine Extensions

This can be divided into two separate cases. The first one models the imaging process and deals with only some special affine transformations, while the second allows for full affine distortions.

3.5.1 Unknown Aspect Ratio

The only distortion likely to happen to an orthographic projection in an actual camera is that the aspect ratio is unknown (the pixel size not perfectly square). This will lead to a skew in the object whenever the object's axis of symmetry is not parallel to either the x- or the y-axis (see figure 3.14).

It is clear that after removing the skew by applying a matrix of transformation

$$\mathtt{T}_S = \begin{pmatrix} s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.37}$$

the aspect ratio will again be 1 and we are in the case discussed above. It is also clear that this method does not work if the axis of symmetry is parallel

to one of the image axes, since then no skew will be introduced through anisotropic scaling.

3.5.2 Full Affine Distortions

Using an affine instead of a weak perspective camera can be modelled[18] by substituting the matrix $\mathtt{T}$ in equation 3.31 with the matrix of an affine transformation

$$\mathtt{T}_A = \begin{pmatrix} t_{11} & t_{12} & t_x \\ t_{21} & t_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.38}$$

with $\det(\mathtt{T}) \neq 0$. This is equivalent to introducing skew and anisotropic scaling in addition to a similarity transform's uniform scaling, rotation and translation.

However, the axis of symmetry is still determined as the line through the bitangent-intersections (intersections are invariant under all transformations discussed here), and it is therefore possible to remove the rotation, translation and uniform scaling by mapping two known points on the axis of symmetry onto two reference points on the y-axis.

This fixes 4 degrees of freedom (2 for the translation and one each for rotation and scaling, out of 6 for an affine transformation — corresponding to the 6 unknowns in equation 3.38) leaving 2: skew and scaling in x-direction.

Skew can easily be removed by finding the transformation

$$\mathtt{T}_{\text{skew}} = \begin{pmatrix} 1 & 0 & 0 \\ A & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.39}$$

this makes the angle between the axis of symmetry and a line through the two bitangent-points on opposite sides of the outline a right angle.

This leaves only the scaling in the x-direction to be solved for. There are two different cases to distinguish:

1. Only the target, but not the reference image, is affinely transformed.
2. Both the target and reference images were affinely transformed.

[18] See chapter 4 for an description of the affine camera.

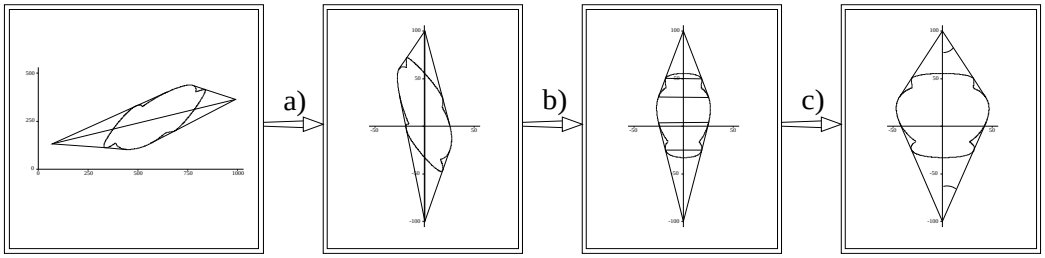


Figure 3.15: *The three steps of removing an affine transformation:*

a) *Removing translation and rotation.*

b) *Removing skew.*

c) *Removing anisotropic scaling.*

An unknown scaling in the x-direction corresponds to a multiplicative factor when taking the tangent of an angle. Equation 3.35 therefore changes to

$$\frac{1}{b}\,\tan(\alpha_o) = \frac{\tan(\alpha)}{\sqrt{\cos^2(\theta) - \sin^2(\theta)\,\tan^2(\alpha)}} \tag{3.40}$$

for case 1, where α is the angle between a bitangent and the axis of symmetry in the reference view, α_o is the corresponding angle in the view that is to be transferred and b is the scale-factor in the x-direction. Given two angles per view it is obviously possible simultaneously to solve two equations 3.40 for both the scaling factor b and the virtual viewing direction θ.

The equivalent equation for case 2 is

$$\frac{1}{b}\,\tan(\alpha_o) = \frac{1/c\,\tan(\alpha)}{\sqrt{\cos^2(\theta) - \sin^2(\theta)\,1/c^2\,\tan^2(\alpha)}} \tag{3.41}$$

it takes three angles per view to solve for the three unknowns b, c and θ. However, even then, only a numerical solution is possible, and this amounts to solving a constrained nonlinear minimisation[19] problem, which might well be hard to solve. Figure 3.15 shows the three main steps.

Although this shows that it is possible to extend these algorithms for use with an affine camera, this is not the method described in chapter 4. This is due not only to the numerical difficulties described earlier, but also to the fact that the number of surfaces producing three usable bitangents over a wide range of viewing directions is very small.

[19]Minimisation instead of root finding, because when using real data, there might very well not be a real solution.

Chapter 4

The Affine Camera

Describe a circle, stroke its back and it turns vicious.

Eugène Ionesco, *The Bald Prima-Donna II*

While the last chapter showed the transfer between two views of the same surface taken with a weak perspective camera, this chapter generalises to an affine projection. Any view (other than a top-view) can be used as a reference view when transferring between two views.[1] Furthermore, a system is introduced that allows the transfer of any view into a so-called *canonical frame*, which allows for the reconstruction of the generating function (where the surface wasn't self occluded) up to anisotropic scaling.

The transfer is accomplished using one conic and one additional point on the axis of symmetry (the implementation uses the intersection of a bitangent pair). The conic used in the examples is the surface's ending contour, but it is of course equally possible to use any other distinguished conic, such as a sharp edge or even a stripe of paint, thus broadening the class of surfaces to which this algorithm can be applied.

The method is introduced by first describing the underlying geometry. This is followed by a discussion of implementational details and some typical results.

4.1 Theoretical Background

4.1.1 The Affine Camera

The affine camera is defined by analogy to the definition of the weak perspective camera in section 3.1.2, with the difference that the matrix of projection `P` in equation 3.8 can now be any affine projection

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & p_{34} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix} \tag{4.1}$$

in homogeneous coordinates (compare e.g. [16]) where $\mathbf{X} = (X_1, X_2, X_3, X_4)^T$ is the real world point $[(X, Y, Z)^T = \frac{1}{X_4}(X_1, X_2, X_3)^T]$ and $\mathbf{x} = (x_1, x_2, x_3)^T$ is the image point $(x, y)^T = \frac{1}{x_3}(x_1, x_2)^T$.

However, for practical purposes this is equivalent to an orthographic projec-

[1] The implementation does not allow for fronto-parallel views. This is due to the fact that the conic in a fronto-parallel view can not be written in the matrix-form required by the program (4 of the 6 parameters A–F would in general have to be infinity).

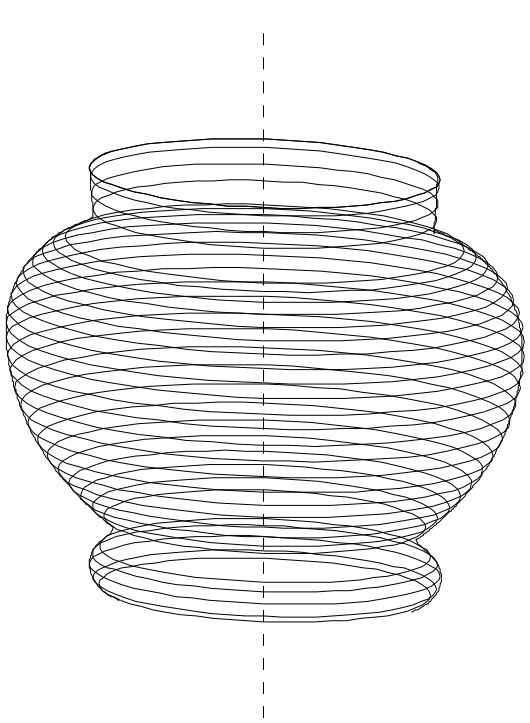


Figure 4.1: *a surface of revolution can be imagined as a stack of circles.*

tion followed by a 2-dimensional affine transformation

$$\mathtt{T}_{\text{aff}} = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ 0 & 0 & 1 \end{pmatrix} \tag{4.2}$$

4.1.2 The Surface's 3D Geometry and its Image

The method of transfer described in this chapter makes use of the fact that a surface of revolution is equivalent to a stack of an infinite number of circles. The midpoints form the axis of symmetry, and all circles are located in parallel planes; Figures 4.1 and 4.2 show examples.

An affine projection will always transfer an ellipse to another ellipse (unlike a perspective projection which can transfer any conic to any other conic). What is more, the midpoints of an ellipse are invariant to any affine transformation, see [3], as are midpoints in general. This means that the image of the stack of circles forming the surface will be ellipses whose midpoints are all on the axis of symmetry (invariance of midpoints). And since the ratio of length on parallel lines is an invariant under affine transformation (see [3, 14]) it is enough to know how 2 points on the axis of symmetry are

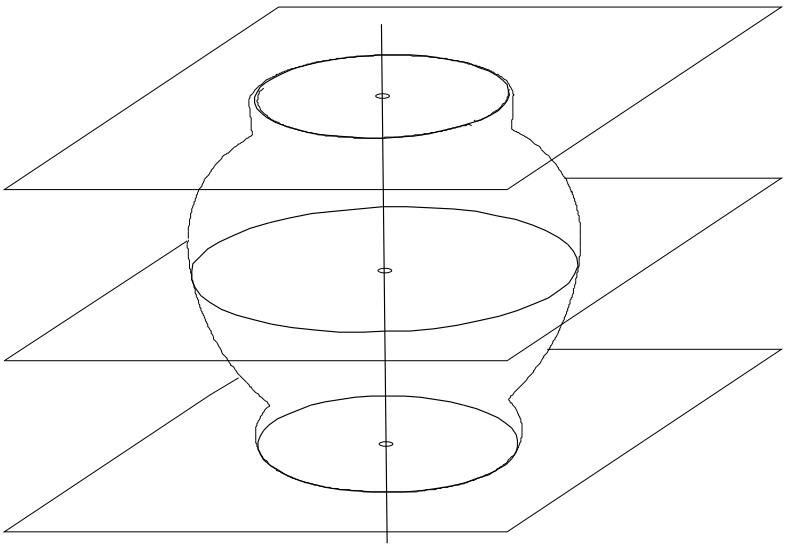


Figure 4.2: *Three of the circles a surface of revolution consists of, all in parallel planes*

transformed to know where any other midpoint is going to.

Circles in parallel planes are projected to ellipses with the same aspect ratio, direction and scaling, i.e. the ratio of two circles diameters before the transformation is the same as the ratio of the two ellipses major or minor axes after transformation. The size of a conic relative to one in a parallel plane is invariant to perspective projection.

The basic idea is that all these are certainly true for an orthographic projection, and an affine projection — an orthographic projection followed by an affine transformation — can only change orientations or scaling globally.

It is therefore possible to determine where each conic's midpoint is projected to if it is known how 2 points on the axis of symmetry[2] are transferred, and it is possible to determine how every single circle is transferred if this is known for one circle. The remaining task is now to find all the conics from the outline.

4.1.3 Acquisition — Calculating the Conics

If the surface of revolution can be thought of as being a stack of circles, each of which can be projected separately onto a plane, then the projected surface's

[2]These could be bitangent-intersections — see section 3 — or the midpoint of a reference conic.

outline will be the envelope of all these (now) ellipses (see figure 4.1). This means that for every outline point there is exactly one ellipse[3] tangent to this point (see [4]). Or, put another way, every tangent to the contour is also tangent to the ellipse containing the contact point.

A conic has 5 degrees of freedom (the midpoint's x and y coordinates, the scale, the aspect ratio and the orientation), 3 of which are fixed once a reference conic is known (all conics have to have the same aspect ratio and orientation, and only one of the midpoint's two coordinates can be freely chosen, since the midpoint has to be on the axis of symmetry).

Requiring the conic to go through a specified outline point will fix one more degrees of freedom, and the condition that the conic has to be tangent to the outline in this point fixes the remaining one. It is therefore always possible to calculate all the conics that have the outline as their envelope (or rather the translation and scaling necessary to transform the reference conic into any of the other conics).

4.1.4 Transfer

Once the conic's midpoint position on the axis of symmetry relative to 2 reference points is known, as well as the conic's scaling relative to a reference conic, it is easy to calculate what this conic would look like viewed from any other direction (see figure 4.3), if it is known how the two reference points on the axis of symmetry and the reference conic transform, since the new midpoint's position relative to the two reference points and the scaling of the new conic relative to the reference conic are both invariant under affine transformation. The envelope of all these conics will then be the transferred outline.

However, it is not even necessary to calculate the envelope of all these conics in order to get the outline. That it is possible directly to calculate the transferred outline point from the conic is shown in the following section.

4.1.4.1 The Tangent Cone

It was shown in section 2 that the intersection between the tangent at the outline point belonging to a known circle and the axis of symmetry is viewpoint independent. This was done using the concept of a *tangent-cone*, that

[3] With the exception only of cusps, for which there a two or possibly even more ellipses tangent to one point.

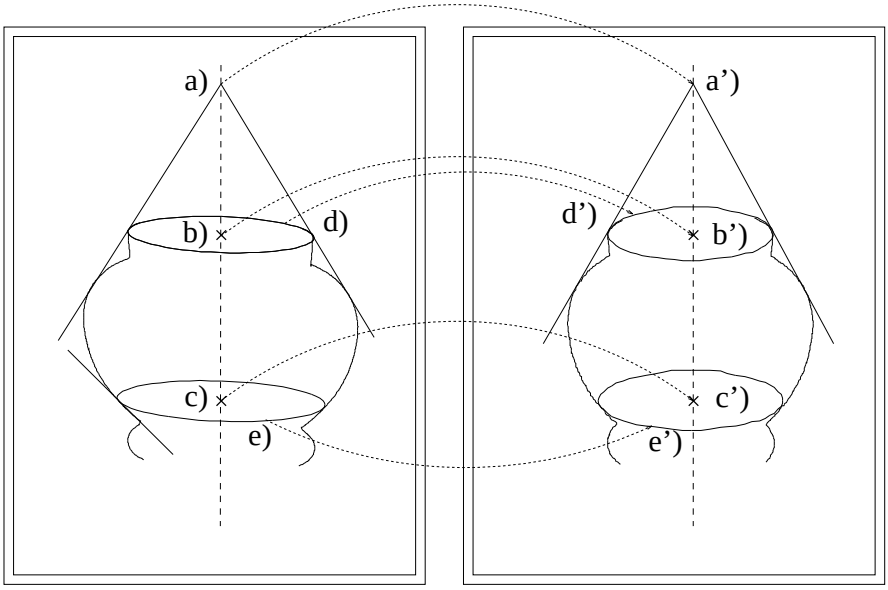


Figure 4.3: *Once it is known how 2 points on the axis of symmetry (a) and (b) are transferred, it is possible to calculate how any other point (c) on the axis of symmetry is transferred. If it is also known how one reference conic (d) is transferred, it is possible to determine the transfer for any other conic (e), a scaled version of (d) (that is, with the same orientation and aspect ratio).*

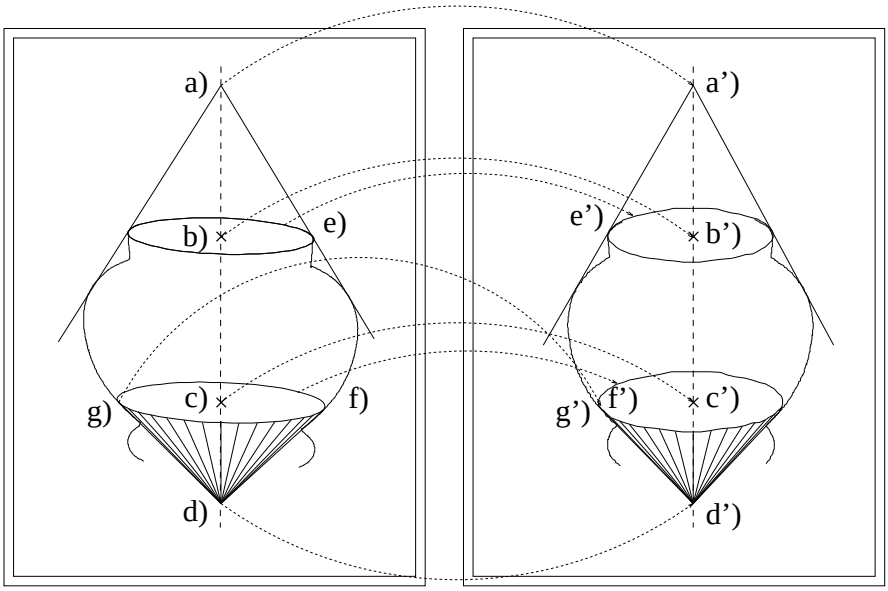


Figure 4.4: *Once it is known how 2 points on the axis of symmetry (a) and (b) are transferred, it is possible to calculate how any other point (c) or (d) on the axis of symmetry is transferred. If it is also known how one reference conic (e) is transferred, it is possible to determine the transfer for any other conic (f), a scaled version of (e). The transferred outline point (g), finally, is the point of tangency between a line through the transferred tangent intersection (d) and the transferred conic (f).*

is, the envelope of all planes tangent to a point both on the surface and on this particular circle.

Transferring the entire tangent-cone belonging to a circle (the cone's base), instead of just the conic, means that it is instantly possible to determine the transferred outline point's position — the point where the base becomes self occluded (see figure 4.4). Another formulation is that this is the point of tangency between a line through the cone's apex and the cone's base.

Still another way to see this is the following: the outline point is the point of tangency between the conic and the outline, and the transferred outline point is therefore the point of tangency between the transferred conic and the transferred outline. The tangent to this contact point will intersect the axis of symmetry in a point that is viewpoint independent. Given the transfer for 2 points on the axis of symmetry, it is possible to transfer this intersection. The point of tangency between a line through the transferred intersection and the transferred conic must then be the transferred outline point.

4.1.5 Summary

4.1.5.1 Prerequisites

The following properties of a surface of revolution are needed for the transfer as described here:

- Two distinguished points on the axis of symmetry, e.g. a conic's midpoint and the intersection of a left and right bitangent (a bitangent-pair).
- One distinguished conic (for example the top conic). This will determine the aspect-ratio and orientation of every other conic.
- The outline to be transferred. It is necessary to know not only the x and y coordinates of every single point, but also the tangent angle (or it must be possible to calculate the outlines tangent-directions).

4.1.5.2 Acquisition

Rather than representing the outline by its x and y coordinates and the tangent angle, it is represented by the conic tangent to the outline at that point and the intersection between the axis of symmetry and a line tangent to that outline point (the cone's apex). The conic can be found as the conic which is tangent to the outline, has its midpoint on the axis of symmetry, and has the same aspect ratio and orientation as the reference conic.

This means it is not necessary to store all the conic's 5 parameters plus the intersection's x and y coordinates, making 7 parameters altogether. Given one reference conic and 2 points on the axis of symmetry, it is possible to store this information in three values per point.

Two possible formats are

(s, m_s, p_s) where s is the conic's scaling relative to the reference conic, m_s is the midpoint's position on the axis of symmetry relative to two distinguished points also on the axis, and p_s is the apex's position on the axis relative to these two points (see figure 4.5). Using the scale factor means that this representation can not be extended for the projective camera, but will only work for the affine case.

(p_1, p_2, p_s) (see figure 4.6) where p_1 and p_2 are the positions of intersections between the axis of symmetry and lines that are tangent to both the

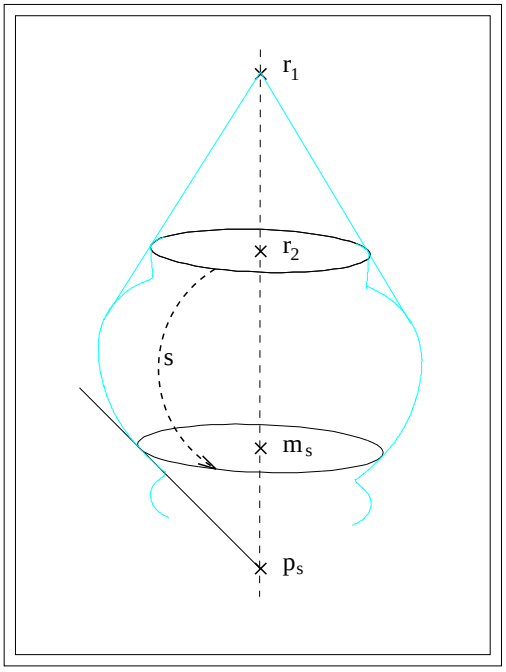


Figure 4.5: *Representing a conic by its scale s relative to the reference conic and its midpoint m_s. p_s is the intersection of the tangent to the outline and the conic with the axis of symmetry. r_1 and r_2 are the two reference points on the axis of symmetry.*

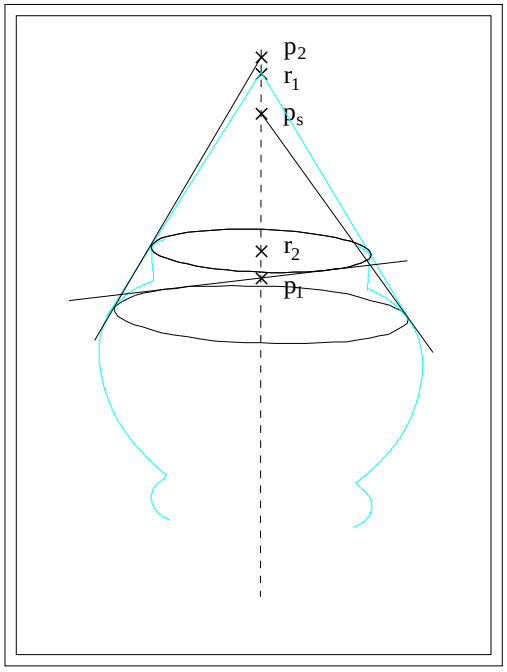


Figure 4.6: *Representing a conic by the two intersections p_1 and p_2 of bitangents between the conic and the reference conic with the axis of symmetry. p_s is the intersection of the tangent to the outline and the conic with the axis of symmetry. r_1 and r_2 are the two reference points on the axis of symmetry.*

reference and the actually used conic. Both positions are again relative to two reference points on the axis. p_s is the same as above. This representation has a projective extension, using 3 points on the axis of symmetry and two conics.

4.1.5.3 Transfer

The outline is now described entirely in terms of conics relative to one reference conic and 2 reference points, either by a scale-factor and two points on the axis of symmetry, or by three points on the axis of symmetry. This description is invariant to an affine transformation, which means that either by choosing one reference conic and two points in a new (canonical) frame, or by identifying the reference conic and points in a second view, it is immediately

possible to determine how all the other conics will transfer.

Once the transferred conics are known, it is possible for each conic to calculate the tangent to the conic through the apex belonging to that conic, thus determining the transferred outline point. (This is really just the reversal of the process used for acquisition.)

4.2 Implementation

Although the acquisition can be done from any view, it is convenient to bring the outline into a common frame prior to acquisition, and transfer by applying a similarity transform such that: the y-axis is the axis of symmetry, the conic's midpoint is at $(\delta, 100)^T$, and the intersection of the first bitangent-pair at $(\delta, -100)^T$; see figure 4.9.

4.2.1 The Common Frame

The transformation into the common frame is done in two steps. First, an Euclidean transformation is calculated such that the sum of the squared and weighted perpendicular distances of all the features — presumed to be on the axis of symmetry — to this axis is minimised, thereby calculating the axis of symmetry. This is followed by a translation and scaling as explained above.

4.2.1.1 The Axis

The calculation of the axis of symmetry is done exactly as in section 3.2.2.1 with the only difference that now the conics midpoint is also used as one of the features that have to lie on the axis of symmetry. The identity matrix is used as the midpoint's covariance matrix. This is shown in figure 4.7.

Figure 4.8 shows the transformed outline.

4.2.1.2 Calculating the Offset and Scale in the y-Direction

A scale in the y-direction and an offset are calculated so that 2 distinct points on the axis of symmetry are moved closest to two fixed points on the y-axis. The frame used is one in which the midpoint of a distinguished conic has

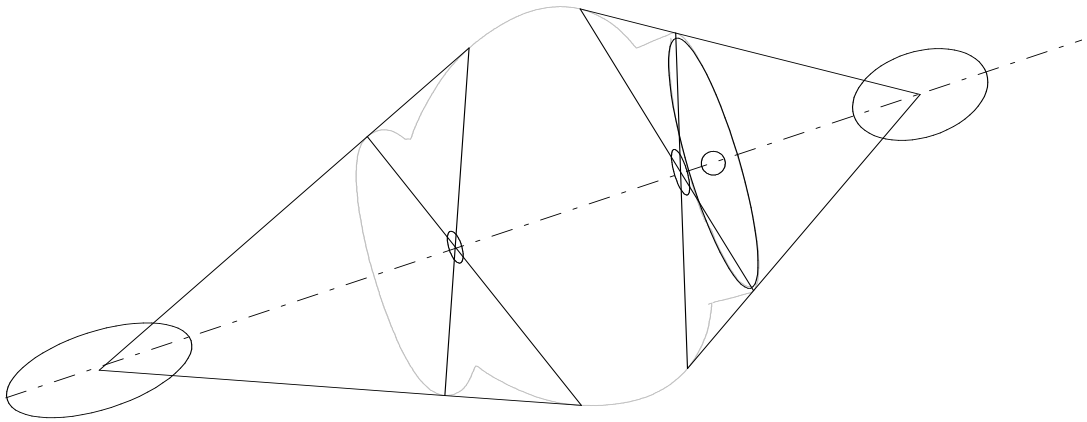


Figure 4.7: *Covariance for intersections and cross-points. The ellipses around the intersections and cross-points denote the maximal error for these intersections, allowing for an error of up to 10 pixels in each tangent-point (see the circle around the conic's midpoint).*

the new coordinates $(\delta, 100)^T$, and in which the first intersection has the coordinates $(\delta, -100)^T$. See figure 4.9.

4.2.2 The Acquisition

The acquisition is done by applying a number of single steps to each outline point. First the orientation of the tangent at this outline point is calculated, together with the tangent's intersection with the axis of symmetry (apex — see figure 4.10a). Next, a conic with the top conic's aspect ratio and orientation is calculated which is tangential to this outline point. Then in the last step, one of the two sets of features mentioned in section 4.1.5.2 is calculated from this conic and the reference conic. Both methods have been implemented.

4.2.2.1 The tangent to the outline

A first approximation of the tangent angle is calculated by finding the regression line through the outline point and its n neighbours. The outline points are then rotated in such a way that the tangent found so far is the

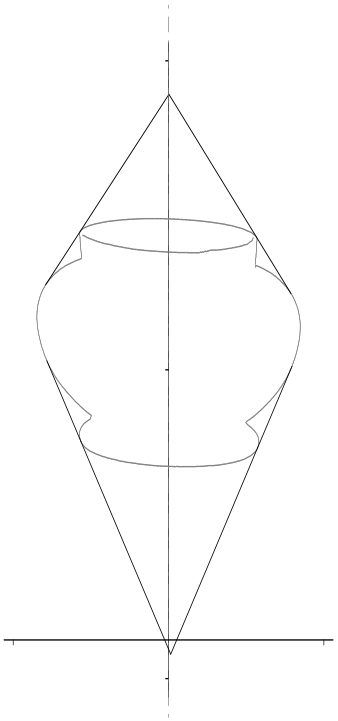


Figure 4.8: *The outline after aligning the axis of symmetry and the y-axis.*

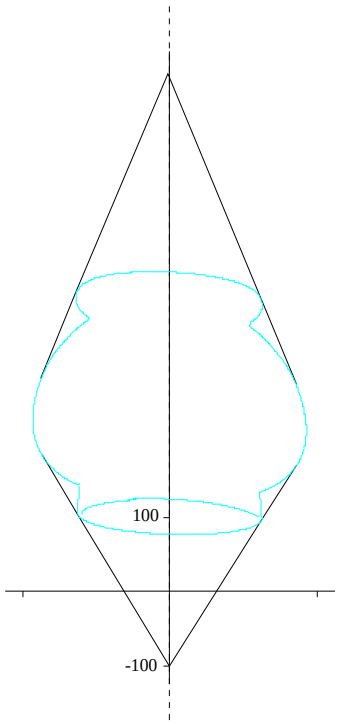


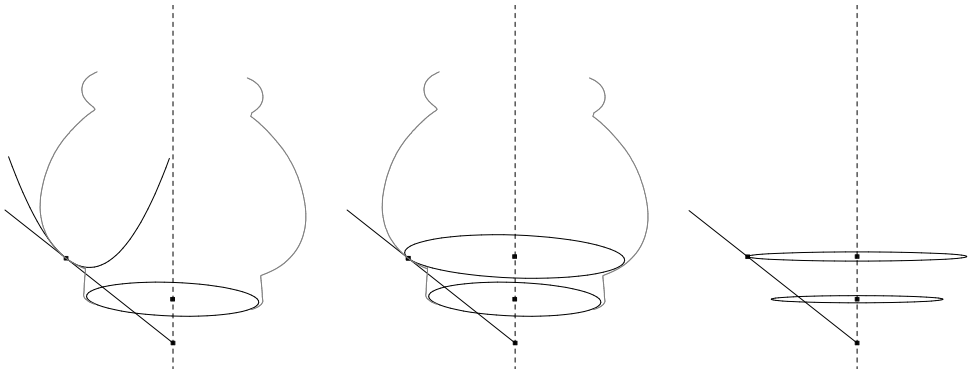
Figure 4.9: *The outline after being transformed into the common frame.*

line $y = \text{const}$, and a least squares quadratic, $y = p_0x^2+p_1x+p_2$, is calculated to find a better approximation for the tangent angle. The algorithm used for calculating the quadratic is described in [17]. The outline's approximation by a quadratic is shown in figure 4.10.a and, in more detail, in figure 4.11.

A simple error measure is calculated ($\sum_i(y_i - y(x_i))^2$), and compared against a threshold (the threshold allows one to influence the accuracy of the fit close to cusps; the lower the threshold, the higher the accuracy used to model the cusps). A new quadratic using $n-2$ neighbouring points is calculated if the threshold is exceeded. The initial value for n is 13 (empirically determined), and there is always an exact solution for $n = 2$ (3 points).

4.2.2.2 The conic tangent to the outline point

How can the conic tangent to the outline point be found? The conic $\tilde{\mathtt{C}}$, tangent at $\mathbf{p}$ and with the tangent orientation $(x_t, y_t)^T$, has to be a scaled



a) The tangent to a point of the outline and it's intersection with the axis of symmetry (apex), calculated by approximating part of the outline around this point with a quadratic.

b) The conic tangent to the outline at this point and with the same aspect ratio and orientation as the reference conic (which is also shown).

c) The transferred point is the contact point between a scaled and translated version of a (new) reference conic — using the scale and translation calculated in b) — and a line through the apex calculated in a).

Figure 4.10: *The 3 steps of transferring an outline.*

and translated version of the ending outline, that is, $\tilde{\mathtt{C}} = \mathtt{T}^{-T}\mathtt{C}\,\mathtt{T}^{-1}$ with

$$\mathtt{T} = \begin{pmatrix} s & 0 & 0 \\ 0 & s & t_y \\ 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

The point $\mathbf{p}$ is an outline point if

$$f = \mathbf{p}\,\mathtt{T}^{-T}\mathtt{C}\,\mathtt{T}^{-1}\mathbf{p} = 0. \tag{4.4}$$

The tangent orientation is

$$\frac{\partial f/\partial x}{\partial f/\partial y} = -\frac{y_t}{x_t}. \tag{4.5}$$

Equation 4.4 is a quadratic whilst 4.5 is a linear equation in 2 unknowns; both s and t_y can be calculated from this. Although one of the equations is a quadratic there is, in general, only one solution (or rather: both solutions are identical). This directly gives the scaling factor s; the new midpoint m_s can easily be calculated from the reference conic's midpoint by applying the matrix $\mathtt{T}$, thus making it easy to find the parameter (s, m_s, p_s) used in the

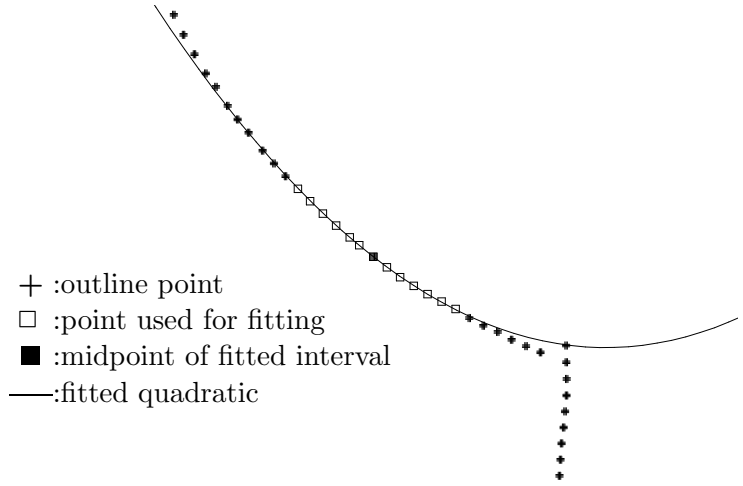


Figure 4.11: *Approximating the outline with a quadratic.*

first representation. The last parameter, p_s, is the intersection between the tangent and the axis of symmetry, it is easily calculated.

To find the bitangents between the conic and the reference conic needed for the second representation, it is necessary to solve a fourth order polynomial. Although these bitangents can be complex, the intersection of the bitangents (on the axis of symmetry) is always real.

4.2.3 Transfer

The only features used to describe the outline are either

1. One scaling factor and two points on the axis of symmetry or
2. Three points on the axis of symmetry.

Once it is known how two points on the axis of symmetry transfer, it is easy to calculate the transfer for any other point, since straight lines remain straight (fixing one degree of freedom), and the ratio of length is invariant under affine transformation (fixing the remaining degree of freedom for a 2D-point). The scaling factor stays constant.

4.2.3.1 Transfer using the first representation

The first representation — using a scaling factor s, the conics midpoint $(0, m_s)^T$, and the apex $(0, p_s)$ — allows for a very easy transfer. The transferred conic's midpoint and the transferred apex are immediately known, and to calculate a scaled version (scaled by the factor s) of the reference conic with this midpoint is trivial. All that remains is to find the point of tangency between this conic and a line through the apex (see figure 1.4.c). This is the intersection between the line $\mathtt{C}\mathbf{p}$ (the polar belonging to a pole $\mathbf{p}$, in homogeneous coordinates) and the conic $\mathtt{C}$, i.e. the solution of the system is

$$(x, y, 1)\,\mathtt{C}\,(0, p_s, 1)^T = 0 \quad (4.6)$$

$$(x, y, 1)\,\mathtt{C}\,(x, y, 1)^T = 0 \quad (4.7)$$

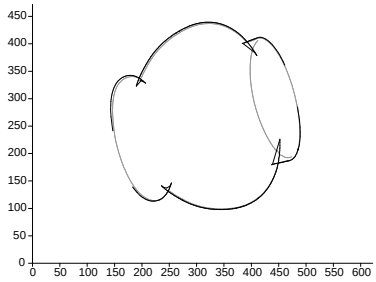
This system has two solutions, belonging to the left and right halves of the outline respectively, and related to each other by the surface's symmetry. However, both sides are transferred separately and the symmetry is not used.

4.2.3.2 Transfer using the second representation

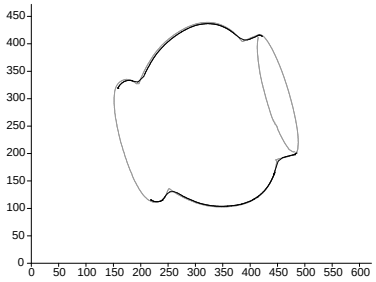
Using the second representation — the apex $(0, p_s)^T$ and the two bitangent-intersections $(0, p_1)^T$ and $(0, p_2)^T$ — makes the transfer a bit more difficult, although more easily adaptable to projective transformation. The three points are again easily transferred. It is then possible to use the mechanism discussed above to find the lines which are tangent to the reference conic, and which go through the points $(0, p_1)^T$ and $(0, p_2)^T$. From there it is possible iteratively to calculate the conic which is tangent to both tangents, which has the same aspect ratio as the reference conic, and whose midpoint is on the axis of symmetry. It is then again possible to calculate the point of tangency between the transferred conic and a line through the transferred apex; this is the transferred outline point.

4.3 Results

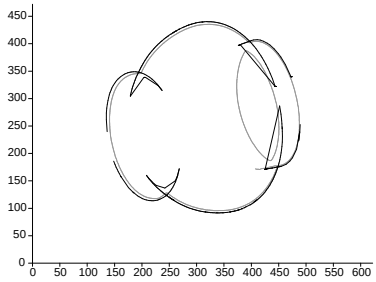
The images in figure 4.12 show some of the results (the results for both methods are identical). The transferred outline is shown in black, with the originals displayed in light grey/dotted for reference. The transfer is surprisingly good, especially when performed from an outline taken under a small angle to one taken under a larger angle (left column), while a loss of accuracy is inevitable when transferring from a larger to a smaller angle (due to self



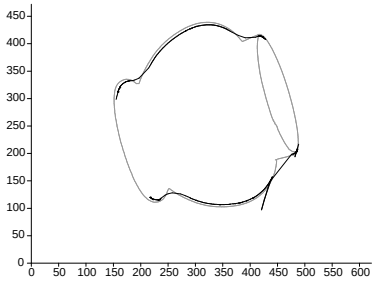
a) Transfer from an roughly 15° view onto an roughly 25° view.



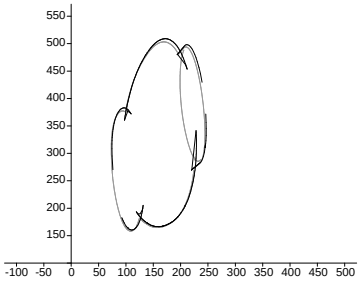
b) Transfer from an roughly 25° view onto an roughly 15° view.



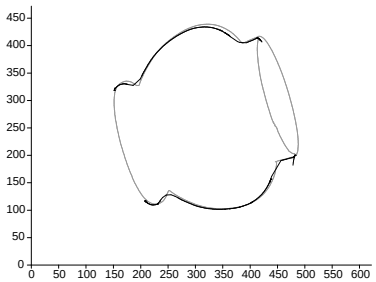
c) Transfer from an roughly 15° view onto an roughly 35° view.



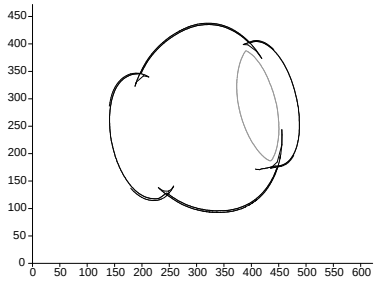
d) Transfer from an roughly 35° view onto an roughly 15° view.



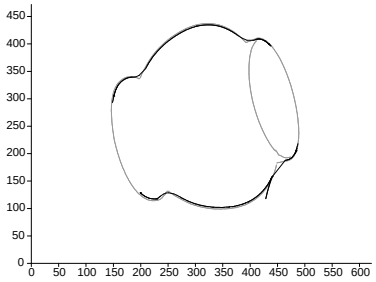
e) Transfer from an roughly 15° view onto an affine transformed roughly 25° view.



f) Transfer from an affine transformed roughly 25° view onto an roughly 15° view.



g) Transfer from an roughly 25° view onto an roughly 35° view.



h) Transfer from an roughly 35° view onto an roughly 25° view.

Figure 4.12: *Transfer using conics*

occlusion), as is shown in the right hand column of figure 4.12 (see especially image d). Note also images e) and f), which demonstrate that the algorithm can indeed cope with affinely transformed outlines.

The spikes at the tops of outlines d), h), and to a lesser extent f), are due to the fact that part of the top conic is becoming transferred too. This leads to characteristic spikes arising from the fact that the conics calculated for this part during the acquisition exceed the outline.

Figure 4.14 shows different views of three different vases, all transferred into one canonical frame. It is obvious from here that recognition using the canonical frame representation is indeed an option, although the agreement of the vases degrades the further away from the top (the bottom side on the image). This is, however, due to badly chosen reference points and can be fixed easily (by choosing reference points of which one is close to the surface's top, and the other close to its bottom), as discussed below.

4.4 Possible Enhancements and Open Questions

4.4.1 Better Features than Intersections

Both methods use a conic's midpoint and a bitangent-pair's intersection as the two features on the axis of symmetry needed to determine where each conic's midpoint is going. This is a rather unfortunate choice for several reasons. The first becomes apparent when looking at figure 4.7; it shows that the intersection's position along the axis of symmetry will in general be less accurate than that of the bitangent-points. The possible error can in fact be orders of magnitude greater than the outline's extension if the bitangents are nearly parallel to the axis of symmetry. It would, therefore, be worthwhile to use a feature which is less prone to errors, for example the midpoint of the second ending outline. (The cross-points, although they too are less error-prone, can not be used for this, since their virtual position on the axis of symmetry can vary.)

Another reason becomes apparent when looking at figure 4.14.

Although the different views of each vase are nearly identical in the vicinity of the top conic (and bitangent-points), their agreement degrades the further from the top they are. This would be improved by choosing a point close to the outline's bottom as the second reference, so that the affine basis encloses

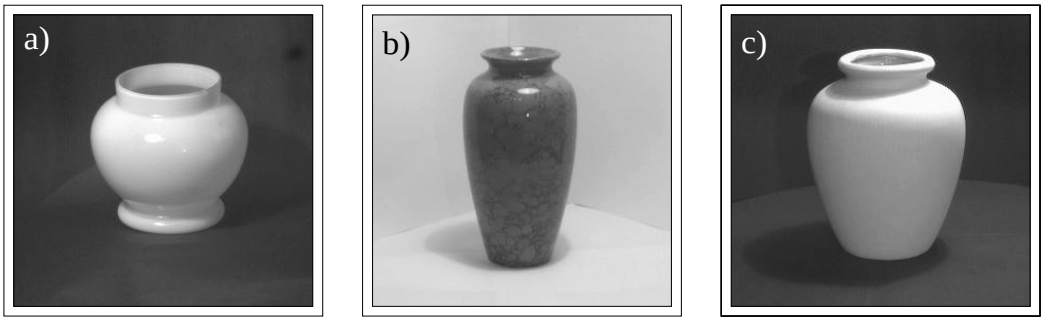


Figure 4.13: *Images of the three vases that were used in the canonical frame representation*
(See figure 4.14)

the outline.

4.4.2 Unused Constraints

Several constraints are still unused, e.g.

- No conic can ever cross the outline at *any* point (since the outline is the conic's envelope).
- The symmetry is not used when transferring outline points.

However, how to make use of these additional constraints is as yet unclear.

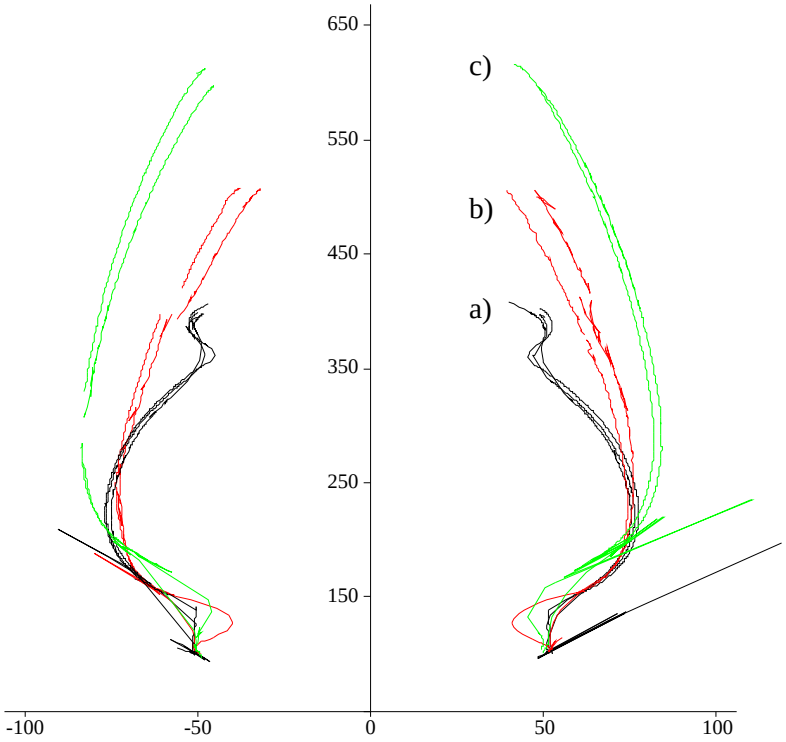


Figure 4.14: *Seven outlines of three different vases, all transferred into the canonical frame.*
Images of the original vases are shown in figure 4.13.

Chapter 5

The Projective Camera

Treat nature in terms of the cylinder, the sphere, the cone, all in perspective.

Paul Cézanne, from Emile Bernard, *Paul Cézanne*

So far methods of transfer for the scaled orthographic (weak perspective) and the affine camera model have been introduced. This chapter presents a method of transfer for which the projective camera is the underlying geometric model. However, this method is based on the one used for the affine camera described in chapter 4. It too allows the transfer from one view onto a second view as well as into a canonical frame, thereby recovering the generating function (where the outline is not self occluded) up to a projective transformation. In contrast to the method for the affine camera that used a conic and a point on the axis of symmetry, this method uses two conics.

The method is introduced by first describing the underlying geometry; this is followed by a discussion of a possible implementation, only part of which (the canonical frame representation) has so far been effected.

5.1 The underlying geometry

5.1.1 The projective Camera

The *projective camera* is defined in analogy to the definitions of the weak perspective camera in 3.1.2 and the affine camera in 4.1.1, but with a matrix of projection `P` (equation 3.8) such that

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix} \tag{5.1}$$

in homogeneous coordinates.

However, it is, for practical purposes, sufficient to think of the projective camera as a *pinhole camera* whose image can than be subject to an arbitrary *projective transformation*, that is a multiplication from the left with the matrix of transformation

$$\mathtt{T}_{\text{proj}} = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & 1 \end{pmatrix} \tag{5.2}$$

The pinhole camera model is a model of a camera in which all rays pass through a focal point generally *not* at infinity. It is also the model used in section 2 (see figure 2.4). Although the focal point is fixed for one view, the image plane can be any plane not passing through the focal point, and it is possible to find a projective transformation to map the current view into any

other view using one of the allowed image planes; this becomes important below.

5.1.2 The Surface's 3D Geometry

The surface is, as in chapter 4, represented by a stack of parallel circles whose midpoints all lie on the axis of symmetry (figure 4.1). In general though, once projected, the resulting ellipses[1] will *not* all have the same aspect ratio nor orientation, as was the case in chapter 4, and the projection of the conics' midpoints will not be identical to the projected conics midpoints. For a general projection, the outline will not be symmetrical (see figure 5.1).

However, using the two bitangent-pairs tangent to the two reference conics, it is still possible to find a rotation and translation such that the y-axis is the axis of symmetry (this can be done exactly as described in chapter 3), and another transformation

$$\mathtt{T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a & 0 & 1 \end{pmatrix} \tag{5.3}$$

such that the outline is again symmetric (and all conics have the same orientation);[2] see figure 5.2. This is similar to the method for removing affine distortions used in chapter 3, but removes perspective fanning along the x-axis.

However, the conics' aspect ratio will still be different, and the question is now if it is possible to find a projective transformation such that the aspect ratio will be the same *for all transferred circles* the surface is made up of. It is of course always possible to find a projection such that the two reference conics will have the same aspect ratio, but that this need not mean that all other conics will have the same aspect ratio too becomes apparent when looking at figure 5.3.

However, there is a family of image planes such that all the projected circles will not only have the same aspect ratio but will indeed be circles.[3] These are the planes that are perpendicular to the axis of symmetry, and in particular the plane the surface is "standing" on. It is obvious that the surface's bottom

[1] Although a projective transformation can transform any conic into any other conic, for example a circle into a hyperbola, this is not of any practical consequence.

[2] Another approach is to map four bitangent points (two on each side) onto four points in a frame in which the points are symmetric around the y-axis.

[3] The two cases are related only by a scaling in the direction of the axis of symmetry anyhow.

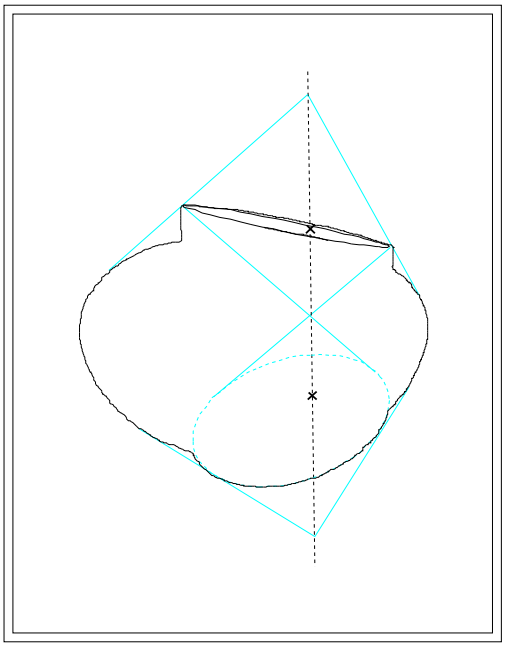


Figure 5.1: *A general projection. Note that*

- *Conics won't have the same aspect ratio.*
- *Conics won't have the same orientation.*
- *The projection of a conic's midpoint will not be identical to the projected conic's midpoint.*
- *The outline is not symmetric.*

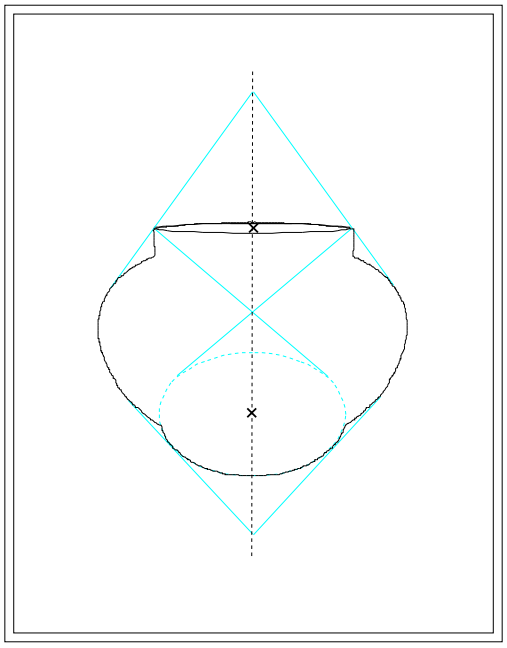


Figure 5.2: *The outline from the left figure made symmetric around the y-axis.*

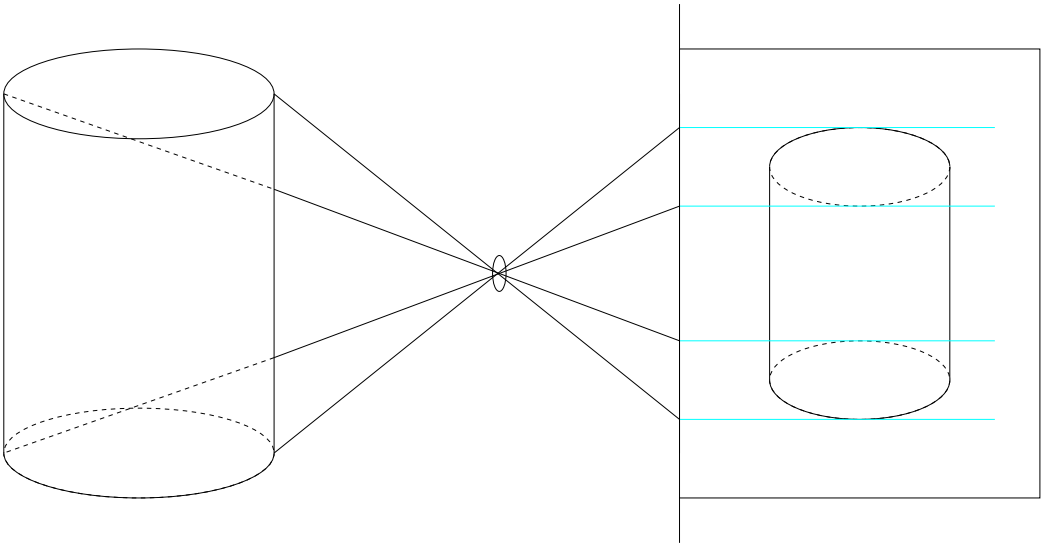


Figure 5.3: *View of a cylinder. The focal point is on the same height as the middle of the cylinder. Although both the top and bottom conics have the same aspect ratio, this is not true for any other conic.*

circle will then be identical to its projection (since it is already on the plane) and will therefore stay a circle. The same is true for all other circles due to the fact that the image plane is parallel to all circles (although all circles will of course have a different scaling). This is shown for a cylinder in figure 5.4;[4] it is true for any focal point outside the surface.

The projective transformation that changes to this image plane — perspective fanning along the y-axis — can be found by numerically solving for the transformation

$$\mathtt{T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & b & 1 \end{pmatrix} \tag{5.4}$$

that makes both conics (the top and bottom one) have the same aspect ratio. However, this has four possible solutions and care is necessary to pick the right one. It is then possible to find an anisotropic scaling in either the x- or y-direction which will transform the conics into circles (although this is not really necessary).

Now all conics have the same aspect-ratio. What is more, it is clear from the

[4]Two different cases can be distinguished. In the first case the viewpoint is below the bottom or higher than the top conic (this is the case shown in most examples), while in the second case the viewpoint's height is between that of the bottom and the top conic — which results in somehow "strange" looking projections. The method discussed here will, however, work for both cases.

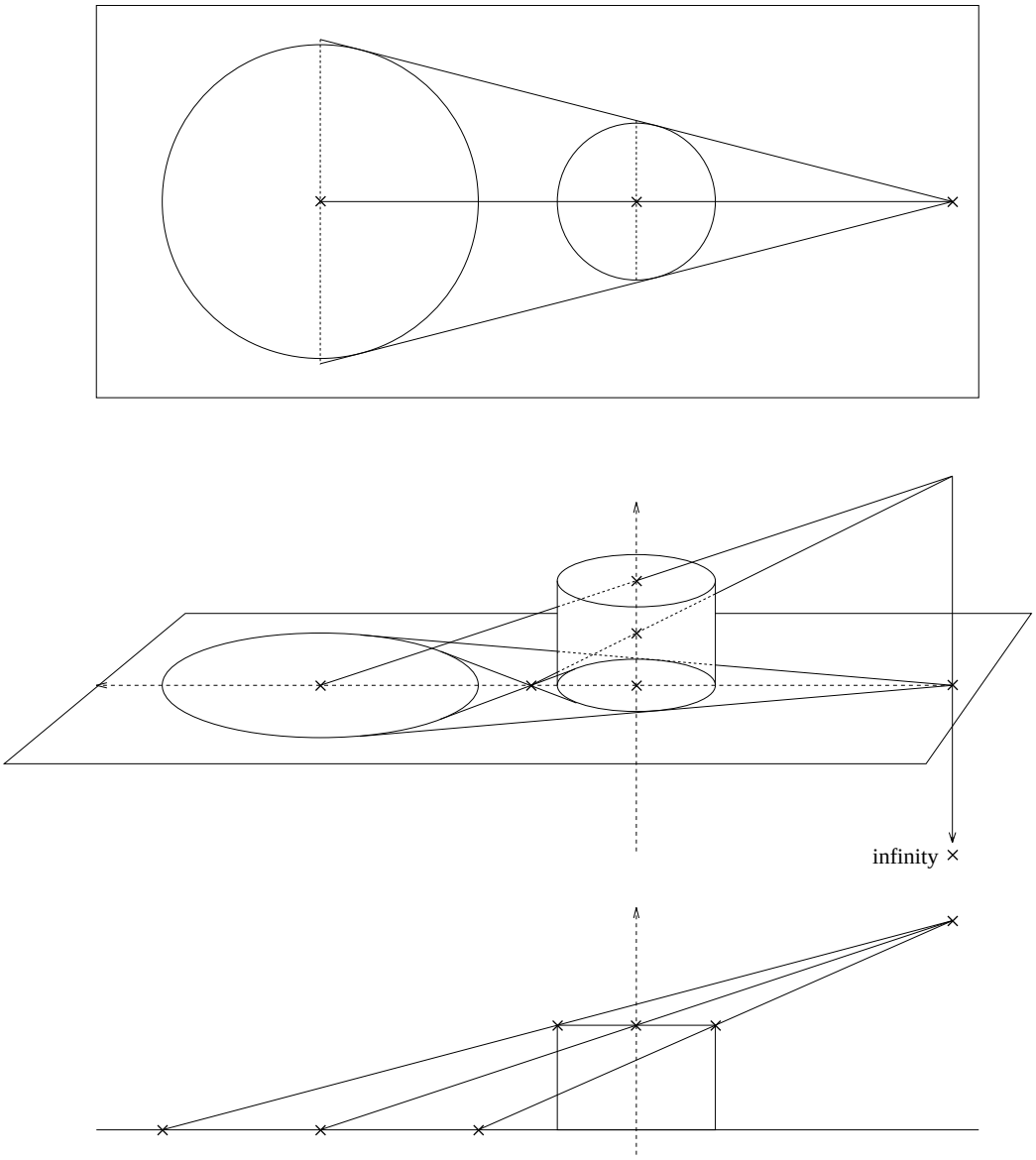


Figure 5.4: *Projecting the circles of which a surface of revolution is made onto a parallel plane will again result in circles. This is viewpoint independent. That a conic's midpoint is invariant to this particular projection can be seen from similarity-triangles. The image shows a top, orthographic and side view of a cylinder and its projection onto the plane it is standing on.*

above construction that the transformed conics' midpoints are identical with the circles' transformed midpoints. That is, it is again possible to use conic-midpoints (although strictly speaking not necessary). All that is needed to find a method of transfer, is to know how points on the axis of symmetry will transform.

5.1.2.1 The Cross-ratio

Given two points on the axis of symmetry in both the reference view and the one to be transferred, it is in the affine case possible to determine where every other point on this axis of symmetry would be transferred to. This was due to the fact that the ratio of lengths on a line (or rather parallel lines) is an invariant under affine transformation. This is no longer true for projective transformation.

The "classic" projective invariant is the *cross-ratio* [17]. Proof of its invariance is covered in most texts about projective geometry (see [22, 24]) and is also found in [16]. It is defined for four points on a line (see figure 5.5) and the simplest definition of the invariant uses ratios of length; for example, if the distance between point A and point B is denoted by $\overline{AB}$ then the cross-ratio τ is

$$\tau = \frac{\overline{AC}}{\overline{BC}} \cdot \frac{\overline{BD}}{\overline{AD}} \tag{5.5}$$

A definition better suited for computing purposes (since one of the points might be at infinity, resulting in the distance being infinity too) uses a homogeneous coordinate system defined on the line and determinants rather than distances:

$$\tau = \frac{|\mathbf{X}_A\mathbf{X}_C|}{|\mathbf{X}_B\mathbf{X}_C|} \cdot \frac{|\mathbf{X}_B\mathbf{X}_D|}{|\mathbf{X}_A\mathbf{X}_D|} \tag{5.6}$$

Here $\mathbf{X}_I$ is the homogeneous coordinate of the point I. The coordinate system can be chosen by arbitrarily fixing an origin and expressing all points by their distance to the origin in the first and 1 in the second coordinate (a point at infinity is denoted by a 0 in the second component). None of the determinants can ever be zero, making the implementation on a computer easy.

The invariance of the cross-ratio with respect to projective transformation means that given 3 points on the axis of symmetry both in the reference view and in the one that is to be transferred, it is possible to determine where every other point on the axis of symmetry is transferred to.

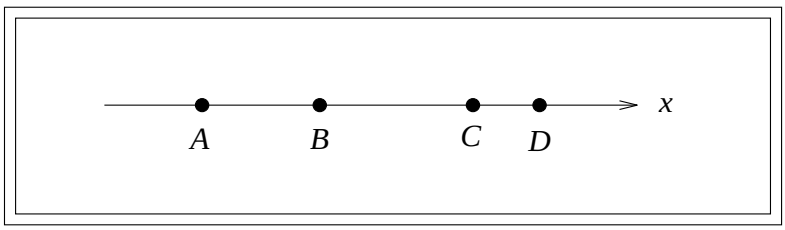


Figure 5.5: *Four collinear points define a simple projective invariant, the cross-ratio*

5.1.3 Summary

It has been shown that it is possible to transform any projective view of a surface of revolution into a frame in which the projection of each cross-sectional circle will have the same aspect ratio, using only two conics (the two ending conics, but it is of course possible to use other conics as well). A second advantage of this frame (apart from all conics having the same aspect ratio) is that the concept of conic midpoints becomes meaningful again.

Given three points on the axis of symmetry (e.g. the reference conic's midpoints and one intersection with the axis of symmetry of a bitangent tangent to both conics) it is possible to transfer outline points in a way analogous to that described in chapter 4, but using the cross-ratio instead of a simple ratio of length. The mechanics of this are described in the following section.

5.2 A possible Implementation

Both the acquisition as well as the transfer described below will assume that we are dealing with outlines already transformed into a frame in which all conics have the same orientation and aspect ratio, and in which the outline is symmetric around the y-axis. That, and how this can be done, has been outlined in section 5.1.

If for some reason it is necessary to effect the transfer onto a real image, it is then always possible to transfer the reference and target view's outlines into the frame first and then reversing the transformation that brought the target view into the frame, but using the reference-view's transferred data.

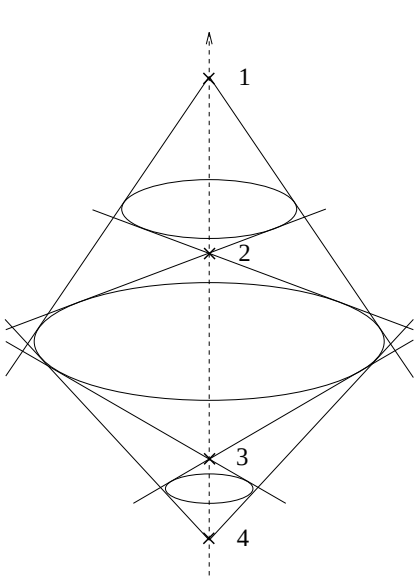


Figure 5.6: *The eight bitangents between an arbitrary conic and the two reference conics, resulting in 4 intersections with the axis of symmetry.*

5.2.1 Acquisition

The acquisition can be done similarly to section 4.2.2 and consists of the following steps:

1. For every single outline point find the orientation of the line tangent to the outline at this point, and the line's intersection with the axis of symmetry (apex).
2. Find the conic that is tangent to the outline in this point, has the same aspect ratio and orientation as the two reference conics, and has its midpoint on the axis of symmetry.
3. In contrast to the affine acquisition, it is now possible to find *four* bitangent-pairs between the conic and the two reference-conics, and consequently also 4 intersections between the bitangent pairs and the axis of symmetry (see figure 5.6).

This gives us *six* points on the axis of symmetry for each conic: the midpoint, 4 bitangent-intersections and the apex. This will be more than enough to find the conic in a second view (the conic will in fact be over-determined).

5.2.2 Transfer

The transfer is also closely related to the one used for affine images. Provided that the target view has also been transferred into the common frame; the transfer can be performed as follows:

1. Transfer all 6 points calculated during the acquisition phase into the frame, using the cross-ratio between each point and three of the following: the two bitangent-intersections from bitangents between the conics, and the two conic midpoints.

2. Find the eight lines (four on each side) that go through the transferred intersections and are tangent to the appropriate conics.

3. Find the conic that is tangent to the eight lines, that has the same aspect ratio and orientation as the target outline's conics and whose midpoint is the transferred midpoint. The conic will be over-determined by this and a least squares fit will be needed to solve for it.

4. Find the tangent to that conic through the transferred apex. The point of tangency is the transferred outline point.

It is also possible to calculate the transfer even onto a target view not in the common frame, if one additional point on the axis of symmetry (maybe an additional bitangent-intersection from bitangents to the outline) is known. This additional point is necessary since midpoints can not be used in an arbitrary view. However, the bitangent-intersections from the bitangents between the two conics can still be used, and this together with the additional point are three points on the axis of symmetry — enough to use the cross-ratio. The transfer can then be done as follows:

1. Transfer the apex and the four intersections using the cross-ratio — the midpoint is not needed.

2. Find the eight lines (four on each side) which go through the transferred intersections and are tangent to the appropriate target-conics.

3. Find the conic that is tangent to the eight lines. It is not possible to use any additional features such as aspect ratio or orientation; however, even with only eight lines, the conic is still over-determined.

4. Find the tangent to that conic through the transferred apex. The point of tangency is the transferred outline point.

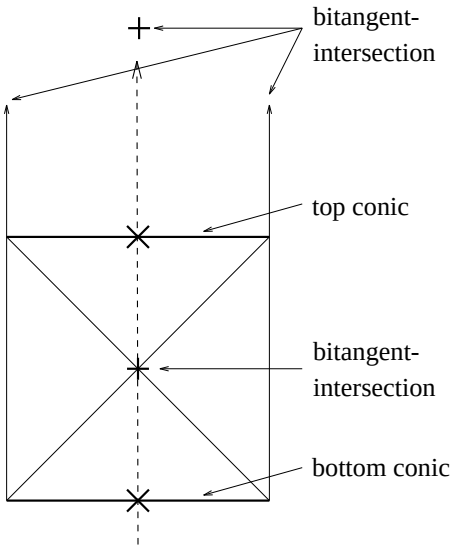


Figure 5.7: *A possible canonical frame. Choosing the reference conics to be lines makes it possible to recover the generating function up to a perspective fanning. One bitangent-intersection is at infinity.*

5.2.3 Transfer into the Canonical Frame

5.2.3.1 How to pick a canonical frame

It has just been outlined how it is possible to transfer from one view in the common frame to any other view, also in the common frame. Therefore all that now remains for transferring into a canonical frame, is to pick one. A possible canonical frame is shown in figure 5.7 but an infinite number of possible frames can be chosen by, for example, fixing the two reference-conics and letting the intersections take care of themselves.

5.3 Results

Figure 5.8 show the results for four views of the same object, all transferred into one common frame. This compares favourably with the results achieved by using the algorithm for the affine case on this — projective — data (figure 5.9. The four outlines are very close to each other and the only differences are due to either self occlusion or a fitted conic that is slightly to small (see figure 5.8). Figure 5.10 shows one of the images used. The different aspect ratio of both the top and bottom conic is very visible.

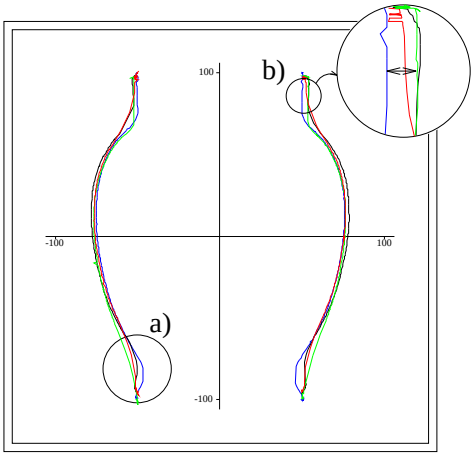


Figure 5.8: *Four outlines of the same vases viewed from different viewpoints, transferred into the canonical frame. Note how well the curves agree. Errors are due to self occlusion (a) or an ill-fitted ending conic (b).*

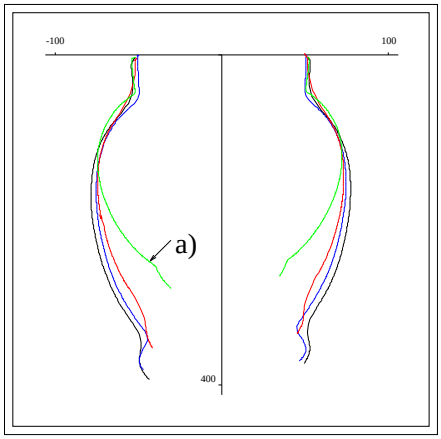


Figure 5.9: *The same outlines, but transferred using the algorithm for affine views. Note how it can't cope with strong perspective distortions (a).*

Figure 5.10: *One of the images whose outline was transferred into the canonical frame. Note the different aspect ratios of the top and bottom conic.*

Chapter 6

Conclusions

Voilà le commencement de la fin.

This is the beginning of the end.

Charles-Maurice de Talleyrand, *At the announcement of Napoleon's defeat at Borodino, 1812*

6.1 A Recognition System

During the last three chapters the transfer from one view of a rotationally symmetric surface onto any other view of the same surface has been discussed for three different camera models. The last two also offered the possibility to transfer the outline into a so-called canonical frame, allowing the recovery of the generating function up to an anisotropic scaling (affine) or perspective transformation. However, how does the transfer fit into the framework of a recognition system?

6.1.1 Transfer between two Views

The transfer from one view onto a second view can be used as a verification step for a hypothesis formed from additional observations (see below). It is possible either to transfer the outline from an image onto a standard representation of the outline, or to transfer the standard representation into the image.

Transferring from the image onto the standard representation has the advantage that the features compared for verification have only to be calculated for the transferred outline, while it is possible to use pre-compiled ones for the standard outline. A multitude of different features are conceivable, ranging from very simple ones such as the distance of n outline points taken in n different directions from an origin (see [17]) or moments (it is possible to use moments even with partly occluded outlines [10]) to very elaborate ones such as descriptions of local curvature.

The advantage of transferring the standard outline into the image is that problems with self-occlusion can be avoided — provided that the standard view is viewed from an angle smaller than the one used in the image (the standard view will preferably be close to a fronto parallel view).

6.1.2 Transfer into a Canonical Frame

This is even more powerful then the simple transfer between views. While the transfer between views is only useful if

1. Very few possible surfaces need to be recognised.
2. A hypothesis as to what the surface is has already been found from another source.

the transfer into the canonical frame allows one to find such a hypothesis. The only hypothesis needed to use the transfer into the canonical frame is that the outline indeed belongs to a surface of revolution; it is then possible to use the canonical-frame representation to take (invariant) measures of the surface which can be used to build an indexing vector which will allow hypotheses generation (see [17]). This hypothesis can then be tested by a model-to-image transfer if necessary.

6.1.3 How to build a Recognition System

A recognition system consists of a lot more than just a verification step or even a module that can generate hypotheses. The input will normally be a grey level image such as in figure 1.2, and algorithms are needed to extract continuous outline curves from there, to find possible candidates for bitangents and conics, to match the bitangents to find possible bitangent-pairs, and to find symmetries (modulo a projectivity or affine transformation) that indicate the presence of rotationally symmetric surfaces.

Fortunately most of this software does already exist. In [17] Charlie Rothwell describes a recognition system which will link up Canny edge data to continuous outlines, and tries to approximate part of the outline with conics or, failing that, straight lines.

He also describes a method for finding bitangents as well as a method for finding projectively equivalent concavities (this was used for assembling a jigsaw puzzle from outline data alone [19]) and can be used for finding projectively symmetric outline parts denoting the possible presence of a rotationally symmetric surface.

All this facilitates finding a number of possible candidates for rotationally symmetric surfaces in an image which can then either be verified or rejected.

6.2 Future Work

The first thing still to be done is obviously to implement the methods described in chapter 5, although there is no reason for doubting the feasibility of this approach.

A problem is, however, the requirement for *two* conics, since most objects tend to have rounded edges, making it difficult precisely to locate conics (see figure 6.1). This makes it interesting to examine how stable the algorithm

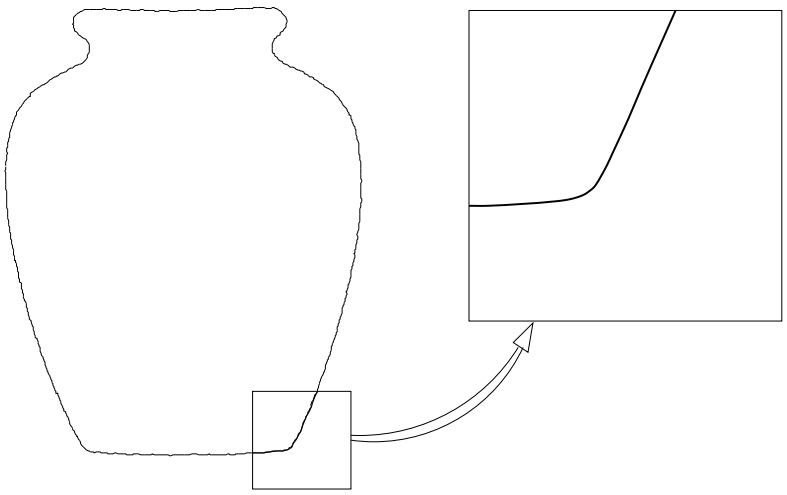


Figure 6.1: *Rounded edges on a surface's generating function make it difficult to find the exact location of the ending conics.*

is regarding errors in the conics. Stability was not a problem for the affine case which only used one conic, but the projective algorithm uses two, which might make errors in the conics more critical.

Also some surfaces do not have *two* distinguished conics, and it might therefore be a good idea to look for an algorithm which does not make explicit use of conics.

Bibliography

[1] K. Arbter, W. E. Snyder, H. Burkhardt, and G. Hirzinger. Application of affine-invariant fourier descriptors to recognition of 3-d objects. *IEEE Trans. Pattern Analysis and Machine Intell.*, 12(7):640–647, July 1990.

[2] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[3] M. Berger. *Geometrie*. Cedic/Fernand Natlan, Paris, 2 edition, 1979.

[4] I. N. Bronstein and K. A. Semendjajew. *Taschenbuch der Mathematik*. G. Grosche and V. Ziegler and D. Ziegler, Harri Deutsch, Thun und Frankfurt (Main), 23th edition, 1987.

[5] J. F. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intell.*, 8(6):679–698, 1986.

[6] R. Cipolla. Active visual inference of surface shape. Technical Report OUEL 1902, University of Oxford, Dept. of Engineering Science, 1991.

[7] M. Dhome, J. T. Lapreste, G. Rives, and M. Richetin. Spatial localization of modelled objects of revolution in monocular perspective vision. In *Proc. 1st Int. Conf. on Computer Vision*, pages 475–485. 1990.

[8] M. P. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, New Jersey, 3rd edition, 1976.

[9] O. D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In G. Sandini, editor, *Proc. 2nd Europ. Conf. on Computer Vision*, volume 588 of *Lecture Notes in Computer Science*, pages 563–578, Berlin Heidelberg, 1992. Springer-Verlag.

[10] A. Fenske. Affininvariante Erkennung überlappender Muster. Interner Bericht 4/90, Technische Universität Hamburg-Harburg, Technische Informatik I, July 1990.

[11] D. A. Forsyth. Recognizing algebraic surfaces from their outlines. In *Proc. 4th Int. Conf. on Computer Vision*, pages 476–480, 1993.

[12] D. A. Forsyth, J. L. Mundy, A. Zisserman, and C. A. Rothwell. Recognising rotationally symmetric surfaces from their outlines. In G. Sandini, editor, *Proc. 2nd Europ. Conf. on Computer Vision*, volume 588 of *Lecture Notes in Computer Science*, pages 639–647, Berlin Heidelberg, 1992. Springer-Verlag.

[13] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 761–764, 1992.

[14] F. Klein. *Elementary Mathematics from an Advanced Standpoint.* Macmillan, New York, 1925.

[15] J. J. Koenderink. *Solid Shape.* The MIT Press, Cambridge, Massachusetts; London, England, 1990.

[16] J. L. Mundy and A. Zisserman. *Geometric Invariance in Computer Vision.* MIT Press, 1992.

[17] C. A. Rothwell. *Recognition Using Projective Invariance.* PhD thesis, University of Oxford, Oxford, U.K., July 1993.

[18] C. A. Rothwell, D. A. Forsyth, A. Zisserman, and J. L. Mundy. Extracting projective information from single views of 3d point sets. In *Proc. 4th Int. Conf. on Computer Vision*, pages 573–582, 1993.

[19] C. A. Rothwell, A. Zisserman, D. A. Forsyth, and J. L. Mundy. Canonical frames for planar object recognition. In G. Sandini, editor, *Proc. 2nd Europ. Conf. on Computer Vision*, volume 588 of *Lecture Notes in Computer Science*, pages 757–772, Berlin Heidelberg, 1992. Springer-Verlag. Also in [16, p.228–251].

[20] C. A. Rothwell, A. Zisserman, J. L. Mundy, and D. A. Forsyth. Efficient model library access by projectively invariant indexing. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 109–114, 1992. Also in [16, p. 398–407].

[21] G. Segal. Geometry of surfaces. undergraduate notes, Nov. 1986.

[22] J. G. Semple and G. T. Kneebone. *Algebraic Projective Geometry.* Clarendon Press, Oxford, U.K., 1952 (reprinted 1979).

[23] C. Singer, E. J. Holmyard, and A. R. Hall, editors. *A History of Technology*, volume 1. Clarendon Press, Oxford, 1954.

[24] C. E. Springer. *Geometry and Analysis of Projective Spaces*. W. H. Freeman and Company, San Francisco and London, 1964.

Index