The Theory and Computation of Digital Surface Curvature

by John Rugis

The Theory and Computation of Digital Surface Curvature
by John Rugis

Introduction

Taken most broadly, this small book combines ideas from the disciplines of differential geometry and 3D computer graphics. The core concepts in differential geometry date back to original work by Gauss in the early 19th century, whereas, of course, computer graphics is a product of the late 20th century.

More specifically, we draw on ideas from disciplines including topology, set theory, complex analysis, computational geometry, and image analysis. We make extensive use of concepts and data structures from computer graphics, especially in computations.

Chapter 1 covers the theory of surface curvature from first principles. Chapter 2 focuses on surface curvature computation given discrete data.

John Rugis
Maraetai, New Zealand 2016.

Contents

Chapter 1

Curvature and Differential Geometry

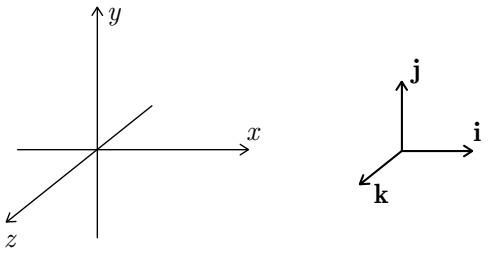
A discussion of both line and surface curvature is a core feature in the subject of differential geometry. See, for example, books (Oprea 2007), (Davies and Samuels 1996), and (do Carmo 1976). Studies on surface curvature can be traced back to original work by Gauss; see, for example, (Pesic 2005). This chapter begins with a review of curvature basics, establishes notational conventions, and adds a few new results (e.g., on n-cuts). The chapter continues with a new scale invariant curvature measure and concludes with a presentation of some curvature visualization examples.

1.1 Notational Conventions

In writing this book, a challenge encountered, because of synthesizing multiple disciplines, was differences in standard notation, sometimes subtle, but none the less significant. We chose to employ vector notation and homogeneous coordinates[1] for both points and directions. Point and directions are thus instances of an abstract vector class in which the fourth coordinate for points is one, and the forth coordinate for directions is zero.

We assign a *right-handed y-up* orthogonal coordinate system to 3D space as shown in Figure 1.1. By convention, the **ijk** *unit basis vectors* align with

[1]The use of homogeneous coordinates facilitates the general use of transformation matrices.



(a) (b)

Figure 1.1: Right-handed y-up coordinate system and the associated unit basis vectors.

the positive direction of the xyz axes respectively.

1.2 Curves and Arcs

We consider *closed simple curves* in 3D space. Such curves can be specified parametrically as a set of points

$$A = \left\{ \mathbf{p}(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} : t_{min} \leq t < t_{max} \right\} \tag{1.1}$$

where $\mathbf{p}(t)$ is a continuous function, $\mathbf{p}(t_1) \neq \mathbf{p}(t_2)$ for all $t_1 \neq t_2$, and $\lim_{(t \to t_{max})} \mathbf{p}(t) = \mathbf{p}(t_{min})$. A point[2] in the curve can be referred to as point $\mathbf{p}$.

Example (a) in Figure 1.2 is a closed, simple curve. Note that the points in (a) form a closed loop. Curve (b) does not meet the $\lim_{(t \to t_{max})}$ criterion and is thus not closed. Curve (c) is self intersecting (i.e., $\mathbf{p}(t_1) = \mathbf{p}(t_2)$ for some $t_1 \neq t_2$) and is thus not simple.

The points in a curve are *ordered* in the sense that if $t_1 < t_2 < t_3$ then $\mathbf{p}(t_1) < \mathbf{p}(t_2) < \mathbf{p}(t_3)$. Thus we can construct sequences of points in a curve as well as consider limits such as $\lim_{\mathbf{p} \to \mathbf{p_0}}$.

A curve is called *smooth* if the function $\mathbf{p}(t)$ is continuously differentiable. We define differentiability at the point $\mathbf{p}(t_{min})$ by considering a

[2] In this work, we use vector notation for both points and directions.

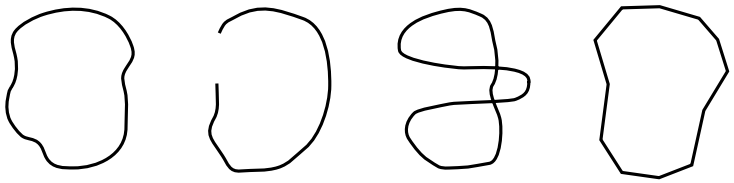


Figure 1.2: (a) Closed, simple, smooth curve. (b) Not closed. (c) Not simple. (d) Not smooth.

shifted parameterization with $(t_{min} - \delta) \leq t < (t_{max} - \delta)$ for some $\delta > 0$, thus extending the domain of the function $\mathbf{p}(t)$, such that $\mathbf{p}(t_{min} - \varepsilon) = \mathbf{p}(t_{max} - \varepsilon)$ whenever $0 < \varepsilon \leq \delta$. Curve (d) in Figure 1.2 contains some corner points and is thus not smooth.

Using the notational convention $\dot{x} = dx/dt$, the *speed* of a curve parameterization at the point $\mathbf{p}$ is given by the following:

$$s(t) = ||\dot{\mathbf{p}}|| = \sqrt{(\dot{x})^2 + (\dot{y})^2 + (\dot{z})^2} \tag{1.2}$$

A curve parameterization $\mathbf{p}(t)$ is said to be *regular* if its speed is never zero. Any point $\mathbf{p}(t)$ at which the speed of a curve parameterization is zero is said to be a *singular point* under that parameterization. Some singularities are removable under a change of parameterization; others are not removable.

By default, in this work, when we refer to a curve, we mean a smooth closed simple curve that has a regular parameterization.

We define an *arc* as any subset of a curve where that subset is parameterized by t with $a \leq t \leq b$ and $t_{min} \leq a < b < t_{max}$. The *end-points* of the arc are $\mathbf{p}(a)$ and $\mathbf{p}(b)$. Arcs that include the point $\mathbf{p}(t_{min})$ can be constructed using a shift in parameterization as given in the previous discussion of differentiabily.

In general, the *arc-length* of an arc is given by the following:

$$\int_a^b s(t)\, dt \tag{1.3}$$

Arc-length can alternatively be considered as a function of t where the length is that from some starting point in a curve, parameterized by t_0, to

an *arbitrary point* in the curve parameterized by t:

$$l(t) = \int_{t_0}^{t} s(t)\, dt \tag{1.4}$$

Note that arc-length is independent of the particular curve parameterization that is used. We will additionally restrict our consideration of curves to only those not containing infinite length arcs. This eliminates so-called *space filling curves*[3].

The following is a basic theorem from differential geometry.

Theorem 1 *A regular curve reparameterized by arc-length has unit speed at all points.*

This reparameterization can be written as

$$A = \left\{ \mathbf{p}\big(t(l)\big) = \begin{bmatrix} x\big(t(l)\big) \\ y\big(t(l)\big) \\ z\big(t(l)\big) \end{bmatrix} : 0 \leq l < l(t_{max}) \right\} \tag{1.5}$$

Note that, to write this reparameterization explicitly, we need to solve Equation (1.4) for t, and this is not always possible.

Definition 1 *The (unit length)* tangent direction vector *to a curve at point* $\mathbf{p}$ *is given by the following:*

$$\hat{\mathbf{t}} = \frac{\dot{\mathbf{p}}}{||\dot{\mathbf{p}}||} \tag{1.6}$$

There is a unique *tangent line* associated with each point $\mathbf{p}_0$ on a curve. The tangent line is the set of points given by

$$L = \left\{ \mathbf{p}(t) = \mathbf{p}_0 + \hat{\mathbf{t}}\, t : -\infty < t < +\infty \right\} \tag{1.7}$$

The tangent line to a curve at a given point is independent of the parameterization used.

Whenever a line is specified in the form of Equation (1.7), both a reference point $\mathbf{p}_0$ and a direction vector $\hat{\mathbf{t}}$ are given. Note that, in general, for any specific line (set of points), the reference point $\mathbf{p}_0$ could be any point on the line (one of an infinite number of possibilities) and direction vector could be $\pm\hat{\mathbf{t}}$ (one of exactly two possibilities).

[3] An example is the *Peano curve* which fills the entire unit square.

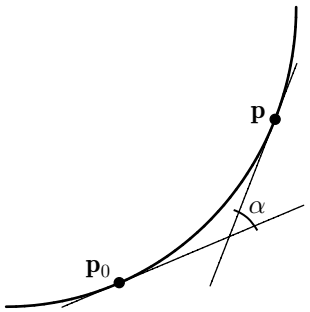


Figure 1.3: Curvature at the point $\mathbf{p}_0$ is defined in the limit $\mathbf{p} \to \mathbf{p}_0$.

Definition 2 *The (unit length)* normal direction vector *to a curve at point* $\mathbf{p}$ *is given by the following:*

$$\hat{\mathbf{n}} = \frac{\dot{\hat{\mathbf{t}}}}{||\dot{\hat{\mathbf{t}}}||} \tag{1.8}$$

Note that the normal vector is undefined both at points where the tangent function is not differentiable and (because of a divide by zero) within any arc where the tangent vector is constant. Also, a basic result from differential geometry shows that, at each point in a curve, the normal vector, where defined, *is orthogonal to the associated tangent vector*.

1.2.1 Curvature of Plane Curves

Even though, in this section, we focus on plane curves, we will work in $\mathbb{R}^3$ because we seek properties that are generally applicable in 3D space.

Definition 3 *Consider a plane curve with arc-length* l *from the point* $\mathbf{p}_0$ *to the point* $\mathbf{p}$*, and the (non-negative) angle* α *between the tangent lines at* $\mathbf{p}_0$ *and* $\mathbf{p}$ *as illustrated, for example, in Figure 1.3. Then the* curvature *of the curve at point* $\mathbf{p}_0$ *is defined to be*

$$\kappa = \lim_{\mathbf{p} \to \mathbf{p}_0} \frac{\alpha}{l} = \frac{d\alpha}{dl} \tag{1.9}$$

Note that curvature as defined here is always non-negative. Since neither the angle between the tangent lines, nor the arc-length, are altered by translation and rotation, we conclude that curvature is a translation and

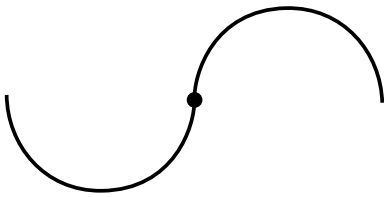


Figure 1.4: Curvature is not defined at the join point of two half circles.

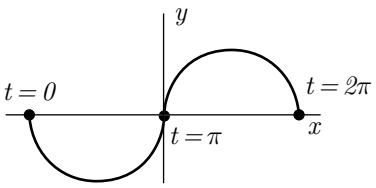


Figure 1.5: A parameterization for an arc constructed from two half circles.

rotation independent property.

Curvature is not necessarily defined at all points on a smooth curve. A well-known result gives constant curvature $1/r$ at all points of a circle having radius r. However, for example, curvature is not defined at the joining point of the arc constructed from two half circles, even when both have the same radius, as shown in Figure 1.4. The existence of curvature at a point is conditional on a parameterization having a continuous second derivative at that point.

We show, in the following example, that the second derivative for this curve is not continuous.

Consider the arc, constructed from joining two unit radius half-circles as shown in Figure 1.5, parameterized by:

$$\mathbf{p}(t) = \begin{cases} 0 \leq t \leq \pi & \begin{bmatrix} \cos(t-\pi) \\ \sin(t-\pi) \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ \\ \pi < t \leq 2\pi & \begin{bmatrix} \cos(-t) \\ \sin(-t) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{cases} \tag{1.10}$$

When $0 \leq t \leq \pi$ we have:

$$\dot{\mathbf{p}}(t) = \begin{bmatrix} -\sin(t-\pi) \\ \cos(t-\pi) \end{bmatrix} \tag{1.11}$$

$$\ddot{\mathbf{p}}(t) = \begin{bmatrix} -\cos(t-\pi) \\ -\sin(t-\pi) \end{bmatrix} \tag{1.12}$$

Making use of the identities $\cos(-t) = \cos(t)$ and $\sin(-t) = -\sin(t)$, we have, when $\pi < t \leq 2\pi$:

$$\dot{\mathbf{p}}(t) = \begin{bmatrix} -\sin(t) \\ -\cos(t) \end{bmatrix} \tag{1.13}$$

$$\ddot{\mathbf{p}}(t) = \begin{bmatrix} -\cos(t) \\ \sin(t) \end{bmatrix} \tag{1.14}$$

When $t = \pi$, both of the first derivatives evaluate to $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Thus, the first derivative of the arc, as a whole, is continuous at the join-point. However, again when $t = \pi$, the second derivatives evaluate to $\begin{bmatrix} -1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ respectively. Since the second derivative of the arc parameterization is not continuous at the join point, curvature is not defined there.

Note that there is also a related discontinuity in the direction of the normal vector $\hat{\mathbf{n}}$ at the join point.

The following theorem from differential geometry summarizes a useful relationship involving curvature.

Theorem 2 *The* Frenet formula,

$$\begin{bmatrix} \dot{\hat{\mathbf{t}}} \\ \dot{\hat{\mathbf{n}}} \end{bmatrix} = \begin{bmatrix} 0 & ks \\ -ks & 0 \end{bmatrix} \begin{bmatrix} \hat{\mathbf{t}} \\ \hat{\mathbf{n}} \end{bmatrix} \tag{1.15}$$

describes the relationship between the curvature, speed, the tangent and the normal at each point in a planar curve.

Since any plane curve in $\mathbb{R}^3$ can be transformed by rotation and translation such that it lies in the xy coordinate plane, it is meaningful, for curvature calculation purposes, to consider xy plane curves.

Curvature for an xy-plane curve can be computed directly, from the curve parameterization, as

$$k = \frac{|\dot{x}\ddot{y} - \ddot{x}\dot{y}|}{(\dot{x}^2 + \dot{y}^2)^{3/2}} \tag{1.16}$$

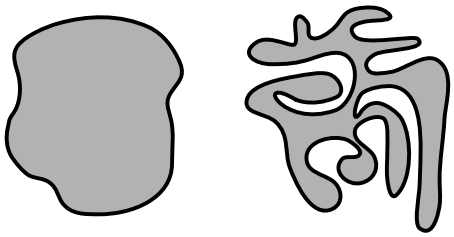


Figure 1.6: Closed simple curves and associated interior regions.

or for a unit speed parameterization as simply

$$k = |\dot{x}\ddot{y} - \ddot{x}\dot{y}| \tag{1.17}$$

Curvature for xy-plane curves, given as a function $y(x)$ in rectangular coordinates, can be computed as

$$k = \frac{|\frac{d^2y}{dx^2}|}{(1 + (\frac{dy}{dx})^2)^{3/2}} \tag{1.18}$$

and for plane curves given as $r(\theta)$ in polar coordinates as

$$k = \frac{r^2 + 2\,(\frac{dr}{d\theta})^2 - r^2\,|\frac{d^2r}{d\theta^2}|}{(r^2 + (\frac{dr}{d\theta})^2)^{3/2}} \tag{1.19}$$

Note that curvature is dependent in general on both first and second order derivatives.

1.2.2 Curvature and Planar Regions

A closed simple planar curve separates the plane into two regions: a bounded *interior* and an unbounded *exterior*. Figure 1.6 shows two curves, each with its associated interior region shaded.

In this work, we associate a (closed simple) planar curve, as well as each of its arcs, with its interior region.

A preliminary definition is required before we reconsider curvature.

Definition 4 *The (planar)* ε-neighborhood *of a point* $\mathbf{p}$ *in a plane is the open disk of all points in the plane whose distance to* $\mathbf{p}$ *is less than* ε.

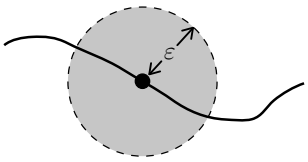


Figure 1.7: A planar ε-neighborhood (open disk).

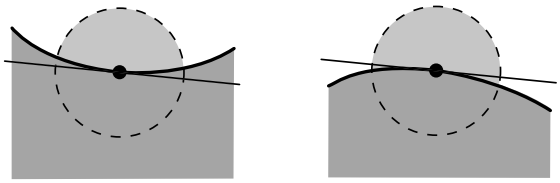


Figure 1.8: (a) Negative curvature. (b) Positive curvature.

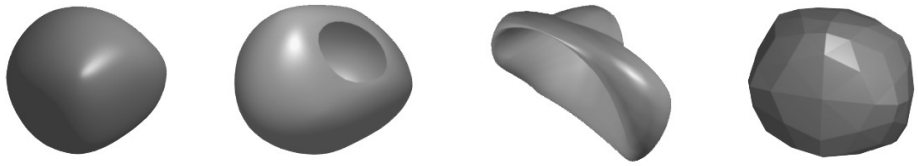
Figure 1.7 illustrates the ε-neighborhood of a point in an arc.

The association of an interior region with a curve allows us to refine our definition of curvature, from Section 1.2.1, with an assignment of a *polarity sign*.

Definition 5 *Consider the non-zero curvature at a point* $\mathbf{p}$ *on a curve and the tangent line at that point. A* negative polarity *is assigned to that curvature if there exists an* ε*-neighborhood of* $\mathbf{p}$ *such that, within that* ε*-neighborhood, all of the points of the tangent line, except for* $\mathbf{p}$ *itself, are part of the associated interior region. The curvature has a* positive polarity *otherwise.*

Figure 1.8 illustrates the definition of curvature polarity. It shows example arcs, interior regions, ε-neighborhoods, and tangents lines for each of the two curvature polarity possibilities.

Throughout the remainder of this book, a polarity will always be assigned to (non-zero) curvature.



(a) (b) (c) (d)

Figure 1.9: (a) Closed, simple, smooth surface. (b) Not closed. (c) Not simple. (d) Not smooth.

1.3 Surfaces and Surface Patches

Consider *closed simple surfaces* in 3D space. Such a surface can be specified parametrically as a set of points

$$S = \left\{ \mathbf{p}(u,v) = \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix} : (u_{min} \leq u < u_{max}) \wedge (v_{min} \leq v < v_{max}) \right\} \tag{1.20}$$

where $\mathbf{p}(u,v)$ is a continuous function, $\mathbf{p}(u_1,v_1) \neq \mathbf{p}(u_2,v_2)$ for all $(u_1,v_1) \neq (u_2,v_2)$, $\forall(v)(\lim_{(u\to u_{max})} \mathbf{p}(u,v) = \mathbf{p}(u_{min},v))$, and $\forall(u)(\lim_{(v\to v_{max})} \mathbf{p}(u,v) = \mathbf{p}(u,v_{min}))$. A point in the surface can be referred to as point $\mathbf{p}$.

Example (a) in Figure 1.9 is a closed, simple surface. Surface (b) has a hole cut in it, so is not closed. Surface (c) can be thought of as having started out like surface (a), but with one side pushed back through the opposite side. So, surface (c), with its self intersection, is not simple, i.e. $\mathbf{p}(u_1,v_1) = \mathbf{p}(u_2,v_2)$ for some $(u_1,v_1) \neq (u_2,v_2)$.

A surface is called *smooth* if the function $\mathbf{p}(u,v)$ is continuously differentiable. We define differentiability at the points $\mathbf{p}(u_{min},v)$ by considering a *shifted* parameterization with $(u_{min} - \delta) \leq u < (u_{max} - \delta)$ for some $\delta > 0$, thus extending the domain of the function $\mathbf{p}(u,v)$, such that $\mathbf{p}(u_{min} - \varepsilon, v) = \mathbf{p}(u_{max} - \varepsilon, v)$ whenever $0 < \varepsilon \leq \delta$. We define differentiability at the points $\mathbf{p}(u,v_{min})$ similarly. Surface (d) in Figure 1.9, composed of triangle facets, is not smooth where the facets meet.

A surface contains families of curves associated with its parameteriza-

tion variables. Consider the family of curves given by

$$A_v = \left\{ \mathbf{p}(u,v) = \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix} : u_{min} \leq u < u_{max} \right\} \tag{1.21}$$

for some finite set of discrete fixed values of v such that $v_{min} \leq v < v_{max}$. Families given by B_u, with a fixed set of values of u, can be constructed similarly. Each point on a surface is contained in two *parameterization curves*, one from each of the possible families A_u and B_v.

Of course, for each curve in the families A_u and B_v, *parameterization speed*, *arc-length*, *regularity*, *singular points*, and *tangent lines* are defined as in Section 1.2.

By default, in this book, when we refer to a surface, we mean a smooth closed simple surface which does not contain any singular points.

We define a (closed) *surface patch* as any subset of a surface where that subset is parameterized by u and v where $a \leq u \leq b$ with $u_{min} \leq a < b < u_{max}$, and $c \leq v \leq d$ with $v_{min} \leq c < d < v_{max}$. Surface patches that include points given $\mathbf{p}(u_{min}, v)$ and $\mathbf{p}(u, v_{min})$ can be constructed using a shift in parameterization as given in the previous discussion of differentiabily.

The *area* of a surface patch is given by the following:

$$\int_a^b \int_c^d s(u,v)\, dvdu \tag{1.22}$$

Note that surface patch area is independent of the particular surface parameterization that is used.

Consider a point in a surface, and the two tangent lines associated, one each, with the two parameterization curves at that point as shown, for example, in Figure 1.10. The unique plane which includes these two tangent lines is called the *surface tangent plane*. Note that, for any given parameterization, the two tangent lines are not necessarily orthogonal to each other.

In general, a plane can be specified as the set of points

$$P = \{\mathbf{p} : (\mathbf{p} - \mathbf{p}_0) \cdot \hat{\mathbf{n}} = 0\} \tag{1.23}$$

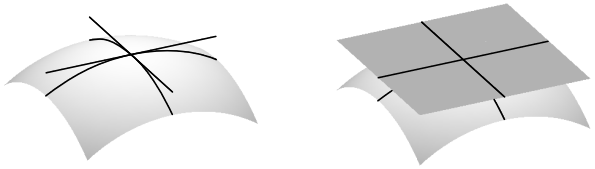


Figure 1.10: Parameterization curves and tangent lines on the left, the tangent plane on the right.

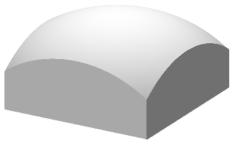


Figure 1.11: Surface patch with a portion of its associated interior region.

where the reference point $\mathbf{p}_0$ can be any point on the plane (one of an infinite number of possibilities) and the *normal direction vector* $\hat{\mathbf{n}}$, could just as well be $-\hat{\mathbf{n}}$ (one of exactly two possibilities).

In the specific case of a surface tangent plane, a normal direction vector can be calculated simply as

$$\hat{\mathbf{n}} = \hat{\mathbf{t}}_\mathbf{1} \times \hat{\mathbf{t}}_\mathbf{2} \tag{1.24}$$

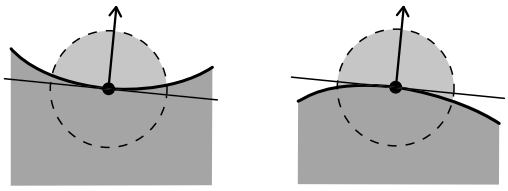
where $\hat{\mathbf{t}}_\mathbf{1}$ and $\hat{\mathbf{t}}_\mathbf{2}$ are surface tangent line directions.

Once we have a surface tangent plane, we can construct a surface tangent line in any direction orthogonal to the tangent plane normal direction. We are thus not limited to the parameterization directions when considering surface tangent lines.

1.3.1 Surface Curvature and Solid Regions

A closed simple surface separates $\mathbb{R}^3$ into two solid regions: a bounded *interior* and an unbounded *exterior*[4]. Figure 1.11 shows a surface patch, with

[4]This is an example of a separation theorem. Separation theorems are important in topology, and have a rich history that dates back to (Jordan 1887) and (Veblen 1905).



(a) (b)

Figure 1.12: Surface patch cross-sections with ε-neighborhood, tangent plane, and the exterior pointing normal.

a portion of its associated interior region shaded.

In this work, we associate a (closed simple) surface, as well as each of its surface patches, with its interior region.

We build up to a definition of the *surface normal vector* with some preliminaries.

Definition 6 *The* ε-neighborhood *of a point* $\mathbf{p}$ *in* $\mathbb{R}^3$ *is the open sphere of all points whose distance to* $\mathbf{p}$ *is less than* ε*.*

A *ray* is the (half-line) set of points given by

$$R = \{\mathbf{p}(t) = \mathbf{p}_0 + \hat{\mathbf{r}}\, t : 0 \leq t < +\infty\} \tag{1.25}$$

where $\mathbf{p}_0$ is the ray starting point and $\hat{\mathbf{r}}$ is the (unit length) ray direction vector.

Consider a point $\mathbf{p}$ on a surface, the tangent plane at that point, and the rays associated with the two possible tangent plane normal vectors. One of those normal vectors is *exterior pointing* in that, for its associated ray, there exists an ε-neighborhood of $\mathbf{p}$ such that, within that ε-neighborhood, all of the points of the ray, except for $\mathbf{p}$ itself, are part of the surface exterior region. See Figure 1.12 for examples.

Definition 7 *The* surface normal vector *at a point in a surface is the exterior pointing tangent plane normal vector at that point.*

A surface *cutting plane* through the point $\mathbf{p}$ in a surface, is any plane, other than the tangent plane, containing $\mathbf{p}$. A *normal cutting plane* through a point in a surface, is a surface cutting plane that includes the surface

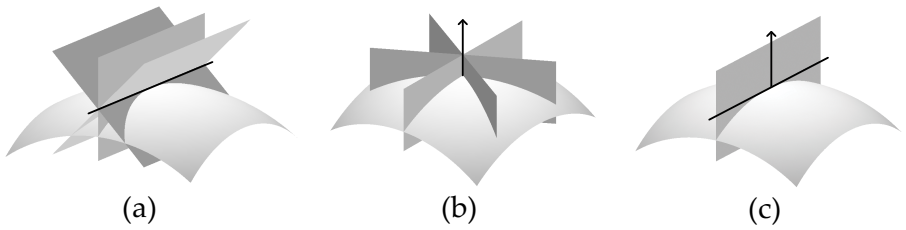


Figure 1.13: Cutting planes: (a) aligned with a surface tangent line, (b) aligned with the surface normal, (c) the unique plane aligned with both a surface tangent line and the surface normal.

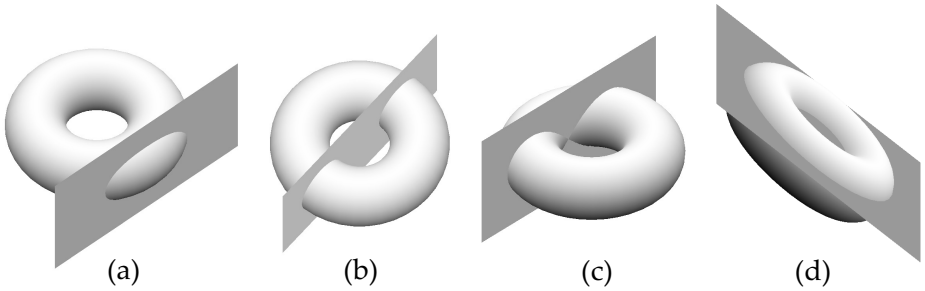


Figure 1.14: Cutting plane intersections: (a) single curve, (b) two curves, (c) self-intersecting curve, (d) concentric curves.

normal at that point. Note that there is a unique normal cutting plane that includes any given surface tangent line. Figure 1.13 illustrates examples of cutting planes.

The intersection of a surface and a surface cutting plane associated with a point in the surface, is (locally) a smooth simple arc which includes that point. Globally, the intersection may include singular points, additional curves, and the curves may self-intersect. Examples are shown in Figure 1.14. Note that in Figure 1.14 (d), the surface interior region is an annulus which lies between two concentric intersection curves.

Definition 8 *The* normal curvature κ_n *at a point in a surface, along the* direction *of a surface tangent line, is the (signed) curvature of the curve which is the intersection of that surface and the normal cutting plane aligned with the tangent line at that point.*

With this definition, curvature polarity is assigned based on surface interior points within the cutting plane subset thought of as planar curve interior points.

Definition 9 *The* principal curvatures, κ_1 *and* κ_2, *at a point in a surface are the respective maximum and minimum values of the normal curvature at that point.*

The following Theorem is a fundamental result from differential geometry and can be found in (do Carmo 1976).

Theorem 3 *The relationship between normal curvature and the principal curvatures is given by*

$$\kappa_n = \kappa_1 \cos^2 \theta + \kappa_2 \sin^2 \theta \tag{1.26}$$

where θ *is the angle between an arbitrary normal curvature (*κ_n*) cutting plane (or direction) and the maximum curvature (*κ_1*) cutting plane (or direction).*

Theorem 3 implies that *the principal curvature cutting planes (and directions) are orthogonal to each other*. (This can be verified by setting θ in Equation (1.26) to the value 0 and then to the value $\pm\pi/2$.)

We are now in a position to define two frequently used standard surface curvature measures.

Definition 10 *The* mean curvature *is defined as*

$$H = \frac{\kappa_1 + \kappa_2}{2} \tag{1.27}$$

Definition 11 *The* Gaussian curvature *is defined as*

$$K = \kappa_1 \kappa_2 \tag{1.28}$$

Often, in practice, as we see later in this book, normal curvatures can be directly calculated, but the principal curvatures cannot. Fortunately, using the following well-known theorem, *mean curvature can be calculated without knowing the principal curvatures*.

Theorem 4 *(Two-Cut Mean Curvature) Using any two normal cutting planes which are orthogonal to each other, the mean curvature at a point on a surface is given by*

$$H = \frac{\kappa_{n_1} + \kappa_{n_2}}{2} \tag{1.29}$$

where κ_{n_1} *and* κ_{n_2} *are normal curvatures associated with those cutting planes.*

Proof. Consider the normal curvatures, κ_{n_1} and κ_{n_2}, associated with two orthogonal normal cutting planes, with the first cutting plane oriented at an arbitrary angle α with the maximum principal curvature direction, and the second cutting plane oriented at the angle $\alpha + \pi/2$. From Equation (1.26) we have that

$$\kappa_{n_1} = \kappa_1 \cos^2 \alpha + \kappa_2 \sin^2 \alpha \tag{1.30}$$

and

$$\begin{aligned} \kappa_{n_2} &= \kappa_1 \big(\cos(\alpha + \pi/2) \big)^2 + \kappa_2 \big(\sin(\alpha + \pi/2) \big)^2 && (1.31)\\ &= \kappa_1 (-\sin \alpha)^2 + \kappa_2 (\cos \alpha)^2 && (1.32)\\ &= \kappa_1 \sin^2 \alpha + \kappa_2 \cos^2 \alpha && (1.33) \end{aligned}$$

Combining Equations (1.30) and (1.33), we have that

$$\begin{aligned} \kappa_{n_1} + \kappa_{n_2} &= (\kappa_1 \sin^2 \alpha + \kappa_2 \cos^2 \alpha) + (\kappa_1 \cos^2 \alpha + \kappa_2 \sin^2 \alpha) && (1.34)\\ &= \kappa_1 \sin^2 \alpha + \kappa_1 \cos^2 \alpha + \kappa_2 \sin^2 \alpha + \kappa_2 \cos^2 \alpha && (1.35)\\ &= \kappa_1 (\sin^2 \alpha + \cos^2 \alpha) + \kappa_2 (\sin^2 \alpha + \cos^2 \alpha) && (1.36)\\ &= \kappa_1 + \kappa_2 && (1.37)\\ & && (1.38) \end{aligned}$$

It follows that

$$\frac{\kappa_{n_1} + \kappa_{n_2}}{2} = \frac{\kappa_1 + \kappa_2}{2} = H \tag{1.39}$$

This proves the theorem. □

In this book, we generalize the two-cut mean curvature theorem to *n-cuts*. But first, some preliminary results are required.

Lemma 1 *For $n \geq 2$, and any non-zero integer c such that n does not divide c,*

$$\sum_{m=0}^{n-1} \cos\big(2c\pi\, m/n\big) = \sum_{m=0}^{n-1} \sin\big(2c\pi\, m/n\big) = 0 \tag{1.40}$$

Proof.

Recall the following identity for the partial sums of a geometric series where z is a complex number (not equal to one):

$$z^0 + z^1 + z^2 + \cdots + z^{n-1} = \frac{z^n - 1}{z - 1} \tag{1.41}$$

Consider the following summation, where $n \geq 2$, and c is a non-zero integer such that n does not divide c:

$$\sum_{m=0}^{n-1} e^{i2c\pi\, m/n} = \left(e^{i2c\pi/n}\right)^0 + \left(e^{i2c\pi/n}\right)^1 + \left(e^{i2c\pi/n}\right)^2 + \cdots + \left(e^{i2c\pi/n}\right)^{n-1} \tag{1.42}$$

$$= \frac{\left(e^{i2c\pi/n}\right)^n - 1}{e^{i2c\pi/n} - 1} \tag{1.43}$$

$$= \frac{e^{i2c\pi} - 1}{e^{i2c\pi/n} - 1} \tag{1.44}$$

$$= \frac{1 - 1}{e^{i2c\pi/n} - 1} \tag{1.45}$$

$$= \frac{0}{e^{i2c\pi/n} - 1} = 0 \tag{1.46}$$

But we also have that

$$\sum_{m=0}^{n-1} e^{i2c\pi\, m/n} = \sum_{m=0}^{n-1} \Big(\cos\big(2c\pi\, m/n\big) + i \sin\big(2c\pi\, m/n\big)\Big) \tag{1.47}$$

$$= \sum_{m=0}^{n-1} \cos\big(2c\pi\, m/n\big) + i \sum_{m=0}^{n-1} \sin\big(2c\pi\, m/n\big) \tag{1.48}$$

Taking Equations (1.46) and (1.48) together gives

$$\sum_{m=0}^{n-1} \cos\big(2c\pi\, m/n\big) + i \sum_{m=0}^{n-1} \sin\big(2c\pi\, m/n\big) = 0 \tag{1.49}$$

which implies that

$$\sum_{m=0}^{n-1} \cos\big(2c\pi\, m/n\big) = \sum_{m=0}^{n-1} \sin\big(2c\pi\, m/n\big) = 0 \tag{1.50}$$

This proves the lemma. □

Lemma 2 *For $n > 2$,*

$$\sum_{m=0}^{n-1} \cos^2\big(2\pi\, m/n\big) = \sum_{m=0}^{n-1} \sin^2\big(2\pi\, m/n\big) = \frac{n}{2} \tag{1.51}$$

Proof.

$$\sum_{m=0}^{n-1} \cos^2\big(2\pi\, m/n\big) = \sum_{m=0}^{n-1} \frac{1+\cos(4\pi\, m/n)}{2} \tag{1.52}$$

$$= \sum_{m=0}^{n-1} \frac{1}{2} + \frac{1}{2}\sum_{m=0}^{n-1} \cos(4\pi\, m/n) \tag{1.53}$$

$$= \frac{n}{2} + \frac{1}{2}\Big(0\Big) = \frac{n}{2} \tag{1.54}$$

$$\sum_{m=0}^{n-1} \sin^2\big(2\pi\, m/n\big) = \sum_{m=0}^{n-1} \frac{1-\cos(4\pi\, m/n)}{2} \tag{1.55}$$

$$= \sum_{m=0}^{n-1} \frac{1}{2} - \frac{1}{2}\sum_{m=0}^{n-1} \cos(4\pi\, m/n) \tag{1.56}$$

$$= \frac{n}{2} - \frac{1}{2}\Big(0\Big) = \frac{n}{2} \tag{1.57}$$

This proves the lemma. □

Theorem 5 *(n-Cut Mean Curvature) Using any $n > 2$ equally spaced (by angle) normal cutting planes, the mean curvature at a point on a surface is the mean of the associated normal curvatures.*

This can be written as:

$$H = \frac{1}{n}\sum_{m=0}^{n-1} \kappa_{n_m} \tag{1.58}$$

where κ_{n_m} is the normal curvature associated with the $(m+1)$-th cutting plane.

Proof. Consider $n > 2$ equally spaced normal cutting planes, with an arbitrary constant angle α between the first cutting plane and the maximum principal curvature direction.

Let $\beta = 2\pi\, m/n$. From Equation (1.26) we have:

$$\sum_{m=0}^{n-1} \kappa_{n_m} = \kappa_1 \sum_{m=0}^{n-1} \cos^2\big(\alpha+\beta\big) + \kappa_2 \sum_{m=0}^{n-1} \sin^2\big(\alpha+\beta\big) \tag{1.59}$$

Consider the κ_1 summation term from Equation (1.59) :

$$\sum_{m=0}^{n-1} \cos^2(\alpha+\beta) = \sum_{m=0}^{n-1} (\cos\alpha\cos\beta - \sin\alpha\sin\beta)^2 \tag{1.60}$$

$$= \sum_{m=0}^{n-1} (\cos^2\alpha\cos^2\beta - 2\cos\alpha\sin\alpha\cos\beta\sin\beta + \sin^2\alpha\sin^2\beta) \tag{1.61}$$

$$= \cos^2\alpha \sum_{m=0}^{n-1} \cos^2\beta - \cos\alpha\sin\alpha \sum_{m=0}^{n-1} \sin 2\beta + \sin^2\alpha \sum_{m=0}^{n-1} \sin^2\beta \tag{1.62}$$

$$= (\cos^2\alpha)\Big(\frac{n}{2}\Big) - (\cos\alpha\sin\alpha)\Big(0\Big) + (\sin^2\alpha)\Big(\frac{n}{2}\Big) \tag{1.63}$$

$$= (\cos^2\alpha + sin^2\alpha)\,\frac{n}{2} \tag{1.64}$$

$$= \frac{n}{2} \tag{1.65}$$

Note that Equation (1.63) employs Lemmas 1 and 2.

Consider the κ_2 summation term from Equation (1.59) :

$$\sum_{m=0}^{n-1} \sin^2(\alpha+\beta) = \sum_{m=0}^{n-1} (\sin\alpha\cos\beta + \cos\alpha\sin\beta)^2 \tag{1.66}$$

$$= \sum_{m=0}^{n-1} (\sin^2\alpha\cos^2\beta + 2\cos\alpha\sin\alpha\cos\beta\sin\beta + \cos^2\alpha\sin^2\beta) \tag{1.67}$$

$$= \sin^2\alpha \sum_{m=0}^{n-1} \cos^2\beta + \cos\alpha\sin\alpha \sum_{m=0}^{n-1} \sin 2\beta + \cos^2\alpha \sum_{m=0}^{n-1} \sin^2\beta \tag{1.68}$$

$$= (\sin^2\alpha)\Big(\frac{n}{2}\Big) + (\cos\alpha\sin\alpha)\Big(0\Big) + (\cos^2\alpha)\Big(\frac{n}{2}\Big) \tag{1.69}$$

$$= (\sin^2\alpha + \cos^2\alpha)\,\frac{n}{2} \tag{1.70}$$

$$= \frac{n}{2} \tag{1.71}$$

Substituting these results back into Equation (1.59) gives:

$$\sum_{m=0}^{n-1} \kappa_{n_m} = \kappa_1\Big(\frac{n}{2}\Big) + \kappa_2\Big(\frac{n}{2}\Big) \tag{1.72}$$

$$= n\Big(\frac{\kappa_1 + \kappa_2}{2}\Big) \tag{1.73}$$

$$\frac{1}{n}\sum_{m=0}^{n-1} \kappa_{n_m} = \frac{\kappa_1 + \kappa_2}{2} = H \tag{1.74}$$

This proves the theorem. □

1.3.2 Surface Curvature Computation

In our discussion of surface curvature so far, we have taken a geometric view in the sense that we have used cutting planes in defining normal curvature and, hence, the principal curvatures. (The motivation for this approach becomes clear later in this book when we consider *digitized surfaces*.) Alternatively, given an explicit surface parameterization (i.e. a set of parametric equations), it is also possible to take an analytical approach to surface curvature and its computation.

To this end, a number of standard working variables are defined in differential geometry. Given a surface parameterization $\mathbf{p}(u, v)$, the *first fundamental form* variables are

$$E = \mathbf{p}_u \cdot \mathbf{p}_u \tag{1.75}$$

$$F = \mathbf{p}_u \cdot \mathbf{p}_v \tag{1.76}$$

$$G = \mathbf{p}_v \cdot \mathbf{p}_v \tag{1.77}$$

and the *second fundamental form* variables are

$$l = \hat{\mathbf{n}} \cdot \mathbf{p}_{uu} \tag{1.78}$$

$$m = \hat{\mathbf{n}} \cdot \mathbf{p}_{uv} \tag{1.79}$$

$$n = \hat{\mathbf{n}} \cdot \mathbf{p}_{vv} \tag{1.80}$$

where we have used the notational convention, for example, $\mathbf{p}_u = \partial\mathbf{p}(u, v)/\partial u$ and $\mathbf{p}_{uu} = \partial\mathbf{p}_u/\partial u$.

The *Weingarten matrix* L is defined as the unique 2×2 matrix, such that, for every point in a surface, we have

$$\begin{bmatrix} \hat{\mathbf{n}}_u & \hat{\mathbf{n}}_v \end{bmatrix} = \begin{bmatrix} \mathbf{p}_u & \mathbf{p}_v \end{bmatrix} L \tag{1.81}$$

where

$$\hat{\mathbf{n}} = \frac{\mathbf{p}_u \times \mathbf{p}_v}{||\mathbf{p}_u \times \mathbf{p}_v||} \tag{1.82}$$

Remarkably, the eigenvalues of the matrix L give the principal curvatures (as $-\kappa_1$ and $-\kappa_2$), and the eigenvectors of L are the *principal curvature directions*. Note that a curvature direction corresponds to the direction of the surface tangent line that is coincident with the associated normal curvature cutting plane.

One of the most important theoretical results in differential geometry is an explicit formulation for the Weingarten matrix in terms of the first and second fundamental form variables:

$$L = \frac{1}{EG - F^2} \begin{bmatrix} Fm - Gl & Fn - Gm \\ Fl - Em & Fm - En \end{bmatrix} \tag{1.83}$$

It is well worth noting that, with the assistance of modern symbolic math computer software that can compute eigenvalues and eigenvectors, the Weingarten matrix can be put to practical use, giving explicit symbolic formulations for the principal curvatures and the principal curvature directions.

Additionally (or alternatively), mean curvature can be given in terms of of the first and second fundamental form variables as

$$H = \frac{En + Gl - 2Fm}{2(EG - F^2)} \tag{1.84}$$

and Gaussian curvature is given as

$$K = \frac{ln - m^2}{EG - F^2} \tag{1.85}$$

And then, using the definitions of mean and Gaussian curvature given in Equations (1.27) and (1.27), we can extract the principal curvatures as

$$\kappa_1 = H + \sqrt{H^2 - K} \tag{1.86}$$

and

$$\kappa_2 = H - \sqrt{H^2 - K} \tag{1.87}$$

For derivations of Equations (1.83), (1.84) and(1.85) see, for example (Davies and Samuels 1996) or (do Carmo 1976).

1.4 Scale Invariant Curvature

It is generally useful to seek *invariant* properties when characterizing 3D objects. At a minimum, translation and rotation invariant characterization is desired as clearly neither of these transformations alters the essential shape property of an object. Surface curvature, a rotation and translation invariant property, meets this requirement.

Any characterization that is additionally *scaling invariant* enables determining the equivalence of shapes independent of size. Perhaps not so obvious, practically speaking, this scale invariance would also enable the use of uncalibrated measurement units in 3D digitization (e.g. scanning).

None of the curvature measures defined so far in this chapter are scale invariant.

In this section we introduce a new scale invariant curvature measure, *similarity curvature*. We also define a similarity curvature space which consists of the set of all possible similarity curvature values.

1.4.1 Geometric Invariants

As previously noted, with existing surface curvature definitions, we already have translation and rotation invariance. What we now seek is scaling invariance. Note that shape characterization based on moments has been studied following work by (Hu 1962), with varying emphasis on invariance with respect to translation, rotation, reflection, or scaling.

Related work (Sapiro 2001), among other things, generalizes and extends the invariance concepts contained in affine differential geometry. Affine invariance is stronger than what we seek in that it includes, for example, squash and stretch transformations.

A number of authors touch on scaling invariant properties in their exploration of multi-scale properties. For example, in (Mokhtarian and Bober 2003), firstly surface feature points such as maximum curvature locations are identified. Then triples of feature points are combined using a geometric hashing algorithm in a way that is scaling invariant. Hash tables for various objects of interest are statistically compared to check for similarity matches between different objects.

1.4.2 A Scale Invariant Curvature Measure

In this section, we present a scale invariant curvature measure that can be assigned *at every point on a surface*. We keep in mind that any definition of scale invariant surface curvature must be related to geometric similarity in which it is well-known that 1) angles are preserved and 2) ratios of lengths are preserved.

We start with some definitions.

Definition 12 *Given principal curvatures κ_1 and κ_2, the curvature ratio κ_3 is defined as*

$$\kappa_3 = \frac{\min(|\kappa_1|, |\kappa_2|)}{\max(|\kappa_1|, |\kappa_2|)}$$

In the case when κ_1 and κ_2 are both equal to zero, κ_3 is defined as being equal to zero. Note that $0 \leq \kappa_3 \leq 1$.

Definition 13 *The similarity curvature R at a point in a surface is defined as*

$$R = \begin{cases} (\kappa_3, 0) & \text{if signs of } \kappa_1 \text{ and } \kappa_2 \text{ are both positive,} \\ (-\kappa_3, 0) & \text{if signs of } \kappa_1 \text{ and } \kappa_2 \text{ are both negative,} \\ (0, \kappa_3) & \text{if signs of } \kappa_1 \text{ and } \kappa_2 \text{ differ, and } \kappa_1 \geq |\kappa_2|, \\ (0, -\kappa_3) & \text{if signs of } \kappa_1 \text{ and } \kappa_2 \text{ differ, and } \kappa_1 < |\kappa_2|. \end{cases} \tag{1.88}$$

Note that $R \in \mathbb{R}^2$.

Theorem 6 *For every closed, simple, smooth surface, the curvature measure R is (positive) scaling invariant.*

Proof. Consider a point on a surface and any associated normal curvature κ, as well as the resultant normal curvature κ' after scaling the surface by a factor s. Since, by definition, $\kappa = d\alpha/dl$, and scaling alters length but not angle, we have $\kappa' = \kappa/s$.
Therefore, after scaling, both of the principal curvatures change by the same factor, and the ratio of the principal curvatures is unchanged. Also, neither the signs, nor the relative magnitudes of the principal curvatures are changed by scaling.

This proves the theorem. □

Henceforth, we will refer to the curvature measure R as the *similarity curvature*.

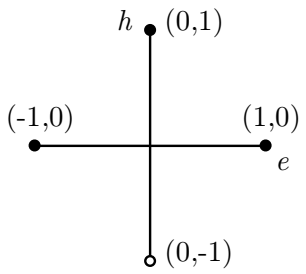


Figure 1.15: eh-plot space for similarity curvature.

1.4.3 The Similarity Curvature Space

Since the set of all possible values of the similarity curvature R is a subset of $\mathbb{R}^2$, it is natural to consider a two-dimensional plot representation. Also recall, from differential geometry, that all surface patches on continuous smooth surfaces are locally either elliptic, hyperbolic (saddle-like), parabolic or planar.

We introduce the similarity curvature plot template in Figure 1.15. The horizontal e-axis is for curvature values at locally elliptic surface points. The vertical h-axis is for curvature values at locally hyperbolic surface points. Both parabolic and planar surface points have similarity curvature value $(0, 0)$ and are thus plotted at the origin. Plotted similarity curvature values will never be off the axes.

When considering examples, it is clear that the similarity curvature at every point on all spheres is constant and equal to $(1, 0)$. The similarity curvature on every planar surface, every cylinder and every cone is constant[5] and equal to $(0, 0)$. Note that this is exactly where the Gaussian curvature is equal to zero.

Continuous motion through points on a smooth surface, taking the similarity curvature at each point, results in a related continuous motion in the similarity plot space. We observe, for example, that it is not possible to traverse from similarity curvature (-1,0) to similarity curvature (1,0) without going through similarity curvature (0,0).

However, it is possible to traverse from (0,1) directly to (0,-1). This can be described by saying that the h-axis *wraps around*. An example where

[5]Excluding the cylinder and cone edges and the cone apex.

this wrapping occurs is given later.

1.5 Curvature Visualization

We can visualize surface curvature magnitude in illustrations by coloring or grey-scale shading the surface points. With this visualization technique, we can see the patterns of curvature variation across a surface. The goal in this section is to gain insight into the nature of, and differences between, curvature measures.

Given an explicit surface parameterization, Equations (1.84) and (1.85) can be used respectively to calculate exact values for mean and Gaussian curvature.

Consider the following sphere parameterization:

$$S = \left\{ \mathbf{p}(u,v) = \begin{bmatrix} r\sin u \\ r\cos u\cos v \\ r\cos u\sin v \end{bmatrix} : (-\pi/2 \leq u < \pi/2) \wedge (0 \leq v < 2\pi) \right\} \tag{1.89}$$

Figure 1.16 shows results for this sphere parameterization when the radius $r = 2$. The color coded sphere object itself is shown on the left, and curvature versus parameterization is shown on the right. For spheres, in general, both the mean curvature and the Gaussian curvature are always constant. For the sphere shown, the mean curvature is $1/2$ and the Gaussian curvature is $1/4$.

Consider the following cylinder parameterization:

$$S = \left\{ \mathbf{p}(u,v) = \begin{bmatrix} hv \\ r\sin u \\ r\cos u \end{bmatrix} : (0 \leq u < 2\pi) \wedge (0 \leq v < 1) \right\} \tag{1.90}$$

Figure 1.17 shows results for this cylinder parameterization when the radius $r = 1$ and the height $h = 2$. For cylinders, in general, both the mean curvature and the Gaussian curvature are always constant, as is the case

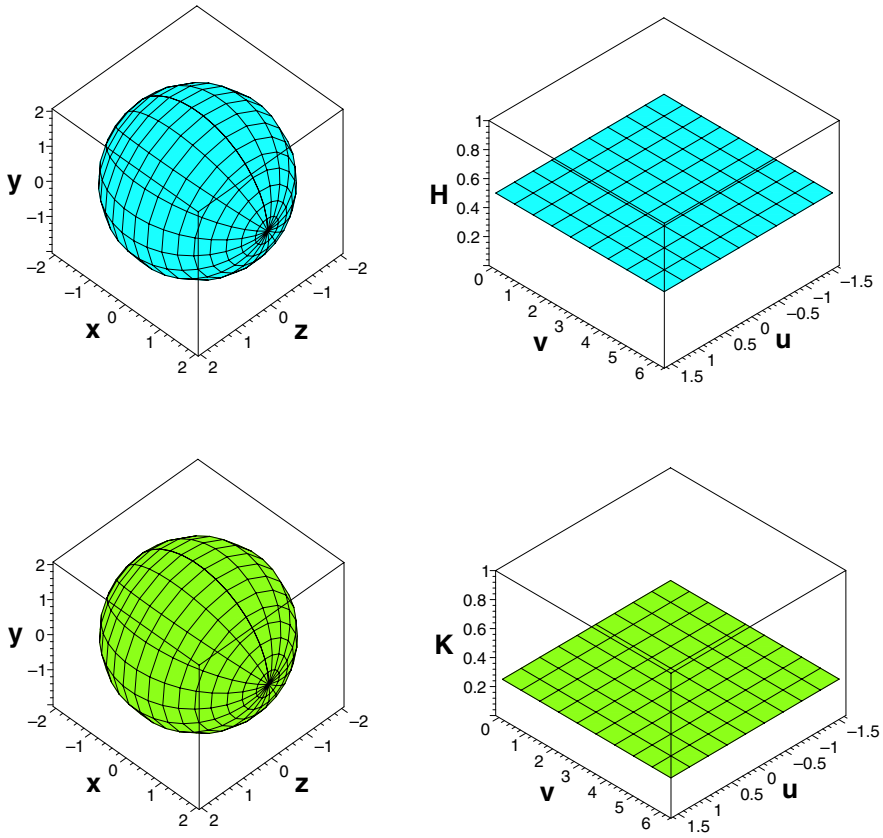


Figure 1.16: Mean (top) and Gaussian (bottom) curvature for a parameterized sphere of radius 2.

for spheres. However, in contrast with spheres, all cylinders have Gaussian curvature zero. For the cylinder shown, the mean curvature is $1/2$.

Consider the following truncated cone parameterization:

$$S = \left\{ \mathbf{p}(u,v) = \begin{bmatrix} hv \\ r(1-v)\sin u \\ r(1-v)\cos u \end{bmatrix} : (0 \le u < 2\pi) \wedge (0 \le v < 0.8) \right\} \tag{1.91}$$

Figure 1.18 shows results for this cone parameterization when the base radius $r = 1/2$ and the height $h = 1$, with the apex cut off at 0.8 times the height. As is the case for cylinders, all cones have Gaussian curvature zero. For the cylinder shown, the mean curvature is 1 at the base, increas-

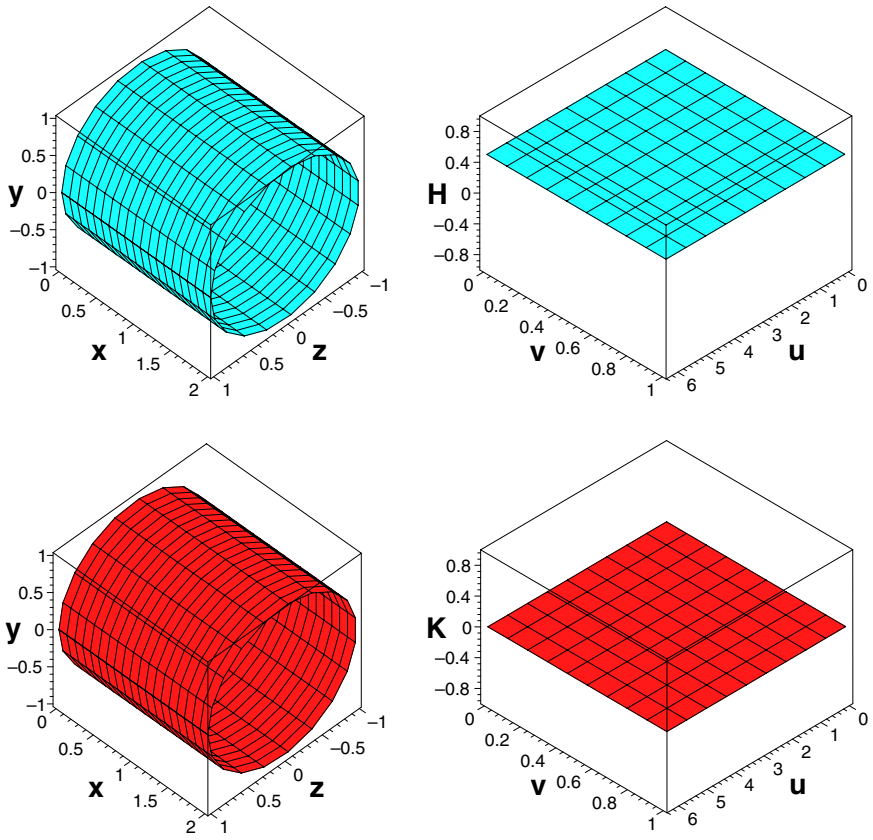


Figure 1.17: Mean (top) and Gaussian (bottom) curvature for a parameterized cylinder of radius 1 and height 2.

ing to 4.5 towards the apex.

Consider the following torus parameterization:

$$S = \left\{ \mathbf{p}(u,v) = \begin{bmatrix} (a + b\cos u)\cos v \\ (a + b\cos u)\sin v \\ b\sin u \end{bmatrix} : (0 \leq u < 2\pi) \wedge (0 \leq v < 2\pi) \right\} \tag{1.92}$$

Figure 1.19 shows results for this torus parameterization when radius $a = 1$ and radius $b = 1/3$. In this figure, the color range for each of the curvatures is separately set to span the range of curvature values. Thus, the range (and offset) of the color scale for each of the curvatures is not the same. However, with this color scale, the curvature visualizations for

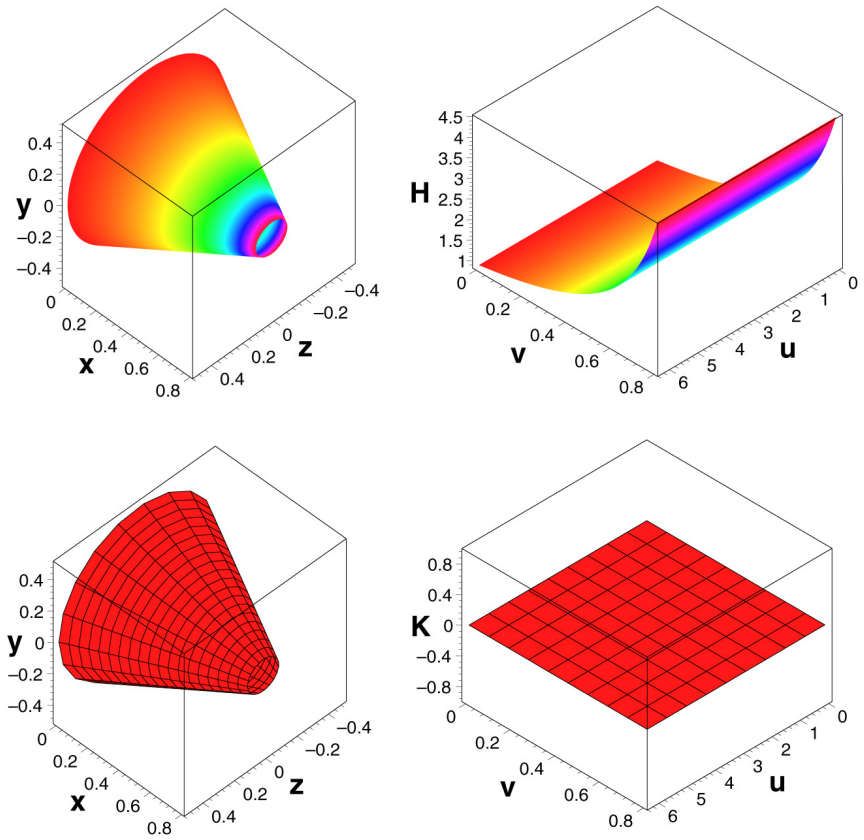


Figure 1.18: Mean (top) and Gaussian (bottom) curvature for a (truncated) parameterized cone having base radius $1/2$ and height 1.

Gauassian and mean curvature appear almost identical. In this sense, at least for the given torus, the *shape* of the curvatures is nearly the same.

Of course, for the given torus, the absolute values of the curvatures are clearly not the same. The mean curvatures are non-negative, while the Gaussian curvatures span both positive and negative values.

Consider the following ellipsoid parameterization:

$$S = \left\{ \mathbf{p}(u,v) = \begin{bmatrix} a\sin u \\ b\cos u\cos v \\ c\cos u\sin v \end{bmatrix} : (-\pi/2 \leq u < \pi/2) \wedge (0 \leq v < 2\pi) \right\} \tag{1.93}$$

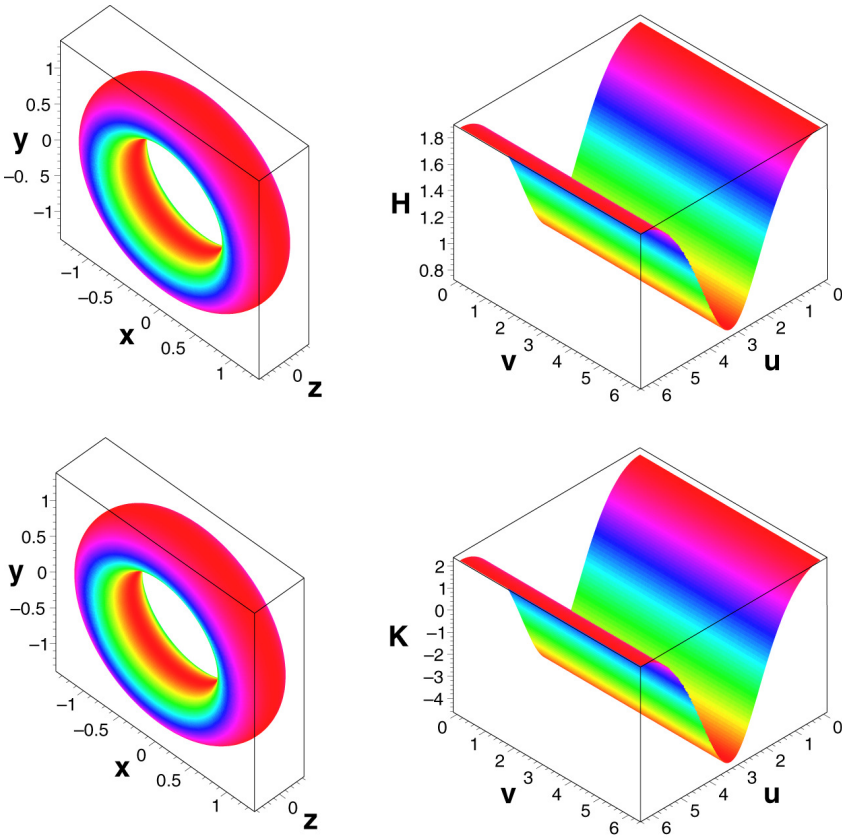


Figure 1.19: Mean (top) and Gaussian (bottom) curvature for a parameterized torus.

Figure 1.20 shows results for this ellipsoid parameterization when $a = \sqrt{3}$, $b = \sqrt{2}$, and $c = 1$. In this figure, the color range for each of the curvatures is separately set to span the range of curvature values. Thus, the range (and offset) of the color coding scale for each of the curvatures is not the same. However, with this color scale, as was the case for the torus, the curvature visualizations for Gaussian and mean curvature appear similar. Again, the *shape* of the curvatures is similar.

Interestingly, at least for the given ellipsoid, the actual values of the Gaussian and mean curvatures are similar as well.

For any given surface prameterization, Equations (1.84) and (1.85) can also be used to develop explicit formulae for Gaussian and mean curvature respectively. However, performing the algebraic operations required to simplify these formulae can be rather tedious!

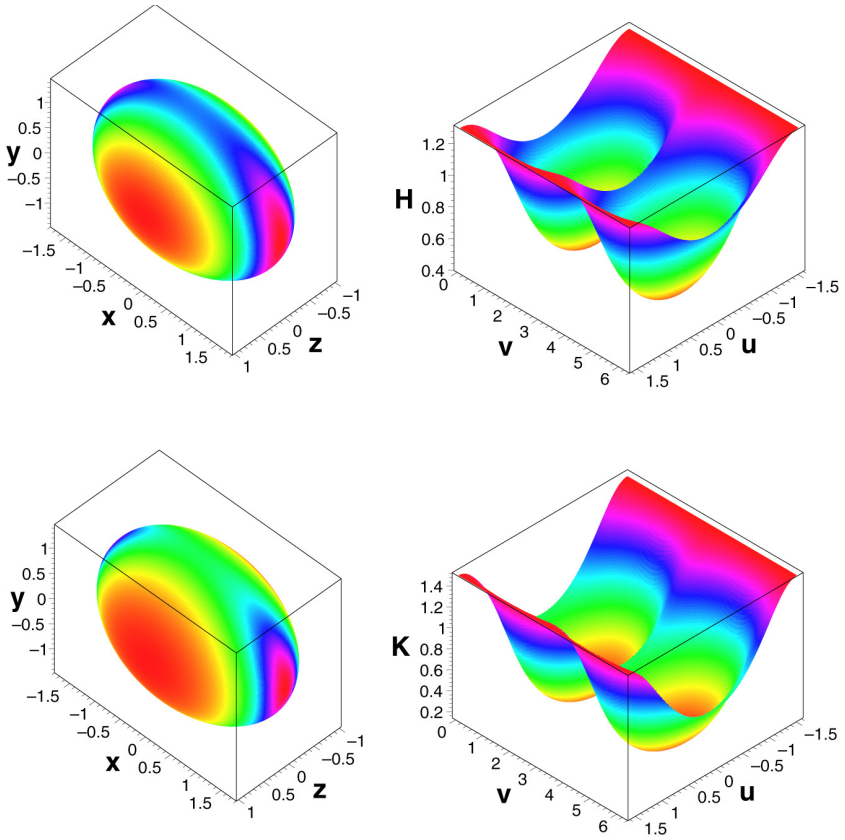


Figure 1.20: Mean (top) and Gaussian (bottom) curvature for a parameterized ellipsoid.

The resultant formula for Gaussian curvature of the ellipsoid given by Equation (1.93) is

$$K = \frac{a^2 b^2 c^2}{(b^2 c^2 - ((b^2 - c^2)a^2 \cos^2 v + b^2(c^2 - a^2)) \cos^2 u)^2} \tag{1.94}$$

and the resultant formula for mean curvature is

$$H = \left(\frac{abc}{2}\right) \frac{b^2 + c^2 + (a^2 - c^2 + (c^2 - b^2)\cos^2 v)\cos^2 u}{(b^2 c^2 - ((b^2 - c^2)a^2 \cos^2 v + b^2(c^2 - a^2)) \cos^2 u)^{3/2}} \tag{1.95}$$

Using the same ellipsoid parameterization, we can also visualize an example of similarity curvature.

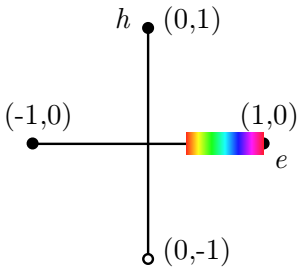


Figure 1.21: EH-plot space color assignment for the ellipsoid example.

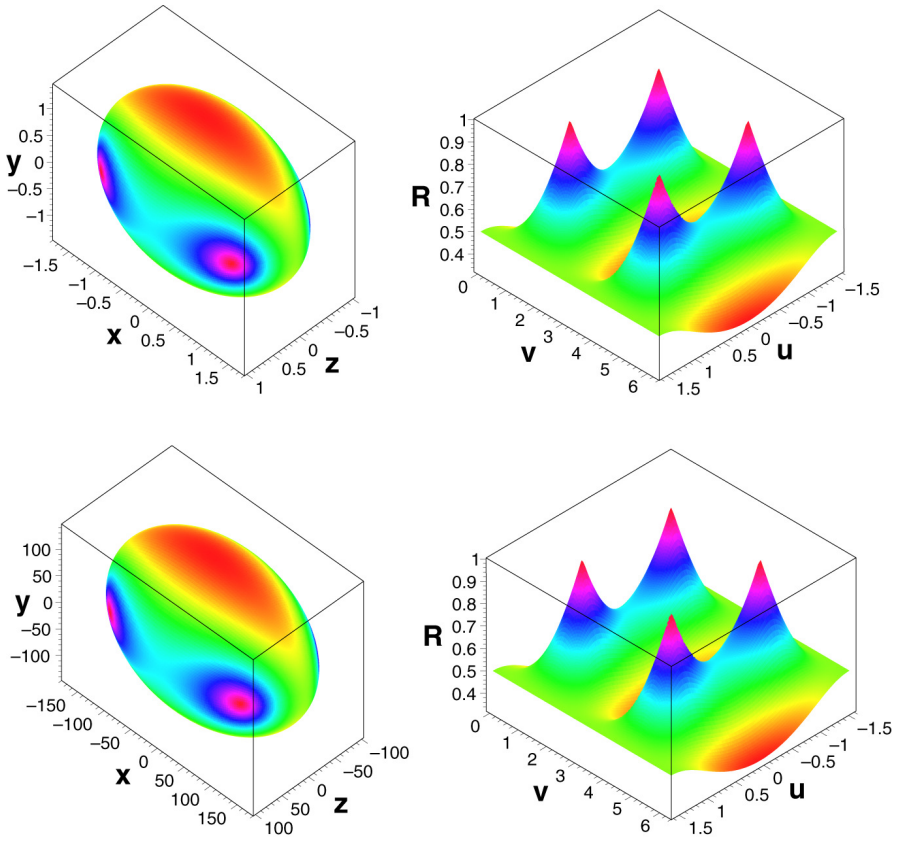


Figure 1.22: Similarity curvature for a parameterized ellipsoid scaled at 1x (top) and 100x (bottom).

Figure 1.22 shows results for this ellipsoid parameterization when $a = \sqrt{3}$, $b = \sqrt{2}$, and $c = 1$ at a 1x scale and a 100x scale. Note that, since similarity curvature for an ellipsoid is always on the e-axis in the similarity curvature space, we have assigned the color range to that axis only as shown in Figure 1.21. As expected, the similarity curvature is the same

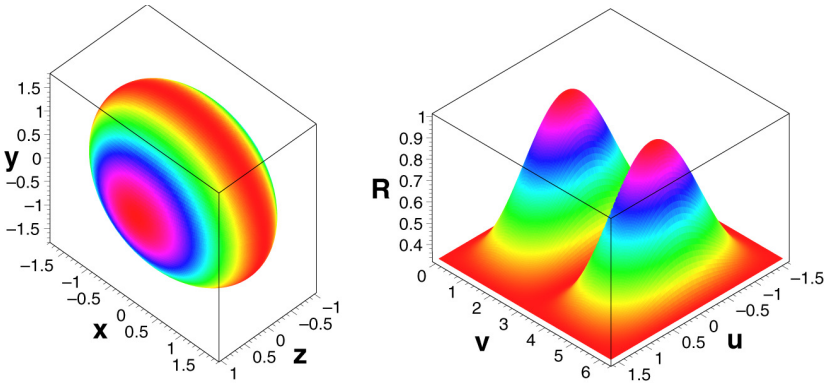


Figure 1.23: Similarity curvature for a parameterized ellipsoid with two equal axes.

at both scales. It is interesting to note that, with this ellipsoid example, similarity curvature takes the sphere curvature value $(1, 0)$ at exactly four points.

Figure 1.23 shows results for the same ellipsoid parameterization when $a = \sqrt{3}$, $b = \sqrt{3}$, and $c = 1$. It is interesting to note that, in comparison with the previous example, since this ellipsoid has two equal length axes, similarity curvature now takes the sphere curvature value $(1, 0)$ only at exactly two points.

Chapter 2

Digital Surface Curvature

The surface curvature of continuous smooth lines and surfaces is a well-defined property, however, when working with point set data, mesh surfaces or 3D digital images, line and surface curvatures can only be estimated (Kalogerakis et al. 2007) (Klette and Rosenfeld 2004) (Mitra and Nguyen 2003). In this chapter we review the nature of digitized surfaces and present a number of curvature estimators, two of which (the 3-cut mean estimator and the similarity curvature estimator) are new.

2.1 Digitized Surfaces

Digitized surfaces arise in numerous practical applications and the form that the digitization takes is highly application dependant. For a range of different applications, see, for example, (Ikeuchi et al. 2003), (Allen et al. 2002), (Levoy et al. 2000), (Bloomenthal 1985), and (Rogers and Satterfield 1980). The data acquisition methods associated with surface digitization can be separated into two categories: volume-based and range-based. Consider, for example, the surface of a spherical object as illustrated in Figure 2.1(a).

Voxel digitizations, as shown, for example, in Figure 2.1(b), are volume-based, with the resultant data-set often being built up from multiple 2D scan slice layers. In volumetric applications, surfaces are usually defined as the boundary associated with a physical density discontinuity within an otherwise solid object (e.g. the medical application of locating bone surface within the volume of an arm). The voxel grid is usually regular, but

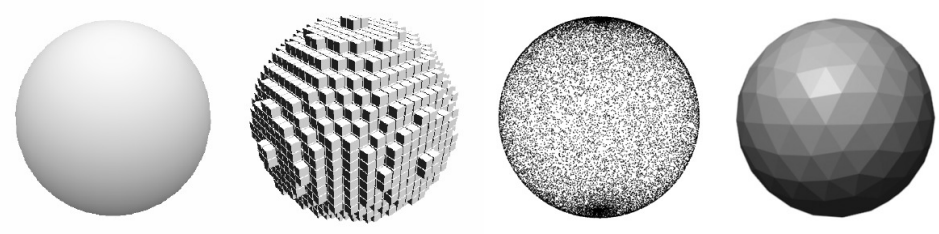


Figure 2.1: (a) Sphere surface. (b) Voxel digitization. (c) Point cloud digitization. (d) Triangle mesh digitization.

the voxels are not necessarily cubes. This situation can arise, for example, from the fact that it is not uncommon for the distance between scan slices to be greater than the resolution within each 2D scan. Voxels are inherently volume elements, and as such, can only estimate actual surface location. See, for example, (Šrámek 1994) and (Gargantini et al. 1993). However, it could be reasonably expected that each voxel would intersect the actual surface. Because voxel data is associated with positions in a fixed grid structure, a voxel data-set can be stored as integers which index position within that grid.

For examples of techniques which employ volumetric data structures, see (Fournier et al. 2006) and(Montani and Scopigno 1990). For a comprehensive theoretical treatment of voxel digitization that draws on concepts from topology, see (Kovalevsky 2008) and (Kovalevsky 2003).

Point cloud digitizations are range-based (e.g., laser triangulation or laser time-of-flight) and the data consists of 3D space points, presented possibly in no particular sequence, and generally with unequal point separation distance. In Figure 2.1(c), for example, the concentration of points is greater at the top and bottom poles of the sphere. Unlike a voxel digitization, each point in a point cloud digitization is considered (ideally) to lie on the actual surface. Point cloud data is usually stored as a collection of real number 3D space coordinates.

For a curvature estimation technique that works directly with point cloud data as input, see (Tong and Tang 2005). Their work builds on previous work in which the concept of *tensor voting* is developed (Tang et al. 1999). For an approach in which the local surface geometry in point cloud

data is modeled by a set of quadratic curves see (Agam and Tang 2005b), (Agam and Tang 2005a) and (Tang and Agam 2004).

Triangle mesh digitizations, as shown, for example, in Figure 2.1(d), approximate an actual surface as a set of adjoining flat triangular surface patches. It seems reasonable to assume that at least some part of each triangle patch would intersect the actual surface, but, in practice, there is no specific general rule. The triangle mesh data structure is often thought of as consisting of *points*, *edges* and *faces*. The face adjacency pattern is an essential characteristic of a triangle mesh, and this adjacency is exploited by many triangle mesh processing algorithms.

Triangle meshes are the standard data structure used in 3D graphics modeling; see, for example, (Foley et al. 2002). 3D graphics modeling is a very active research area resulting in varied and numerous techniques. See, for example, books (Lengyel 2004) and (Hill and Kelley 2007). Triangle meshes can be derived from both voxel based and point cloud digitizations (e.g. (Cooper et al. 2003), (Kobbelt et al. 2001), (Curless 2000), (Curless and Levoy 1996)), and in this sense can be thought of as having higher order structure. Alternatively, a triangle mesh can also arise naturally when a range scan is acquired using a controlled rasterization pattern.

Most real-world digitization processes introduce noticeable *noise* into the acquired data-set. This noise results in some uncertainty as to wether or not, for example, a voxel actually intersects the real surface, or a point in a point cloud actually lies on the real surface. Fortunately, it is possible to mathematically model noise and to reduce its negative impact through *filtering*(e.g. (Rugis 2006), (Rugis and Klette 2006)).

Henceforth, we will refer to the data-set associated with a surface digitization as a *digital surface*.

2.2 Gaussian and Mean Curvature Estimators

All of the curvature estimators presented in this chapter require known point adjacency neighborhood information. This adjacency information is inherent in a triangle mesh[1] and, for convenience, we limit our consider-

[1]For any given set of surface points, more than one triangulation is possible, however, this does not affect the validity of the curvature estimators which are presented.

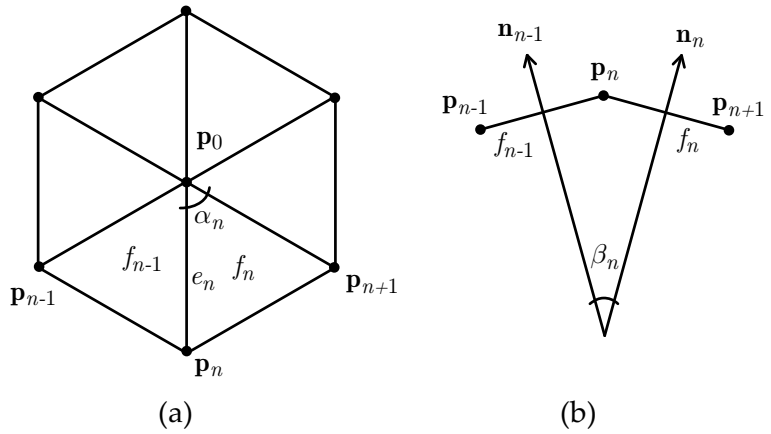


Figure 2.2: Triangle fan surface patch: (a) plan view, (b) cross-section with face normals.

ation to triangle meshes. In practice, this restriction to triangle meshes is not particularly limiting in that conversion from other digitization types is usually possible.

The estimators used in this book are all *local* in the sense that only knowledge of the nearby neighboring points is used in the computation of curvature.

2.2.1 Triangle Umbrella

First we review existing Gaussian and mean curvature estimators used by other authors, for example, (Alboul and van Damme 1996), (Dyn et al. 2000), and (Meek and Walton 2000), which rely on knowledge of a complete *triangle umbrela* neighborhood around a point of interest.

With reference to Figure 2.2(a), consider a point in the surface and, say, six adjacent points. The points are thought to be connected by *edges*, and edges enclose, in this case, six *faces*. We also identify an area $\mathcal{A}(f_n)$ and a central angle α_n associated with each face f_n.

In Figure 2.2(b), we identify a surface normal vector associated with each face from an edge-on view point. The dihedral angle between adjacent face normals is designated as β. Angle β is positive if the faces form a convex surface (when viewed from the exterior) and β is negative if the

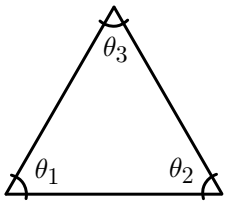


Figure 2.3: The interior angles of an arbitrary triangle.

faces form a concave surface (again, when viewed from the exterior).

The **Gaussian curvature** at the point $\mathbf{p}_0$ is estimated by

$$\tilde{K}(\mathbf{p}_0) = \frac{3(2\pi - \sum \alpha_n)}{\sum \mathcal{A}(f_n)} \tag{2.1}$$

Interestingly, we will show, in the following derivation, that this estimator is generally valid, without change, in the case of any adjacency point count greater than or equal to three.

This estimator can be derived as follows:

Consider, the interior angles of an arbitrary triangle as shown in Figure 2.3. Of course, for a flat triangle, the sum of the interior angles is π, but for a triangle inscribed on a surface, this is not generally true. The well-known Gauss-Bonnet theorem tells us that, for any triangle inscribed on a surface, the *integral Gaussian curvature* (also known as the *total curvature*), over the area of the inscribed triangle, is equal to the interior *angle excess*. This can be expressed as:

$$\int K da = (\theta_1 + \theta_2 + \theta_3) - \pi \tag{2.2}$$

In order to extend this idea to polygons, consider an arbitrary inscribed polygon having $n \geq 3$ sides, and n interior angles ϕ, as illustrated in Figure 2.4(a). This polygon can be subdivided into $(n-2)$ inscribed triangles as shown in Figure 2.4(b). The total curvature over the area of the polygon can be expressed as the sum of total curvatures associated with each of

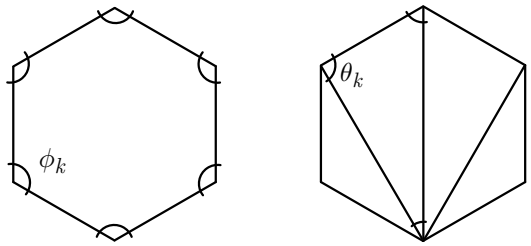


Figure 2.4: An arbitrary polygon: (a) interior angles, (b) triangulated.

those $(n-2)$ triangles:

$$\int K da = \sum_{T=1}^{n-2} \Big((\theta_1 + \theta_2 + \theta_3) - \pi \Big) \tag{2.3}$$

$$= \left(\sum_{T=1}^{n-2} (\theta_1 + \theta_2 + \theta_3) \right) - \Big((n-2)\pi \Big) \tag{2.4}$$

$$= \left(\sum_{T=1}^{n-2} (\theta_1 + \theta_2 + \theta_3) \right) + 2\pi - n\pi \tag{2.5}$$

But, observe that, again with reference to Figure 2.4, the sum of all the interior angles θ in all of the $(n-2)$ triangles is equal to the sum of the n interior angles ϕ of the polygon:

$$\sum_{T=1}^{n-2} (\theta_1 + \theta_2 + \theta_3) = \sum_{k=1}^{n} \phi_k \tag{2.6}$$

Substitution into Equation (2.5) gives

$$\int K da = \left(\sum_{k=1}^{n} \phi_k \right) + 2\pi - n\pi \tag{2.7}$$

Now consider a triangle mesh digitization of the same (arbitrary) inscribed n-sided polygon as illustrated in Figure 2.5. Note that we have separately identified a central angle α associated with each of the n triangles. Because these are flat triangles, we have

$$\sum_{T=1}^{n} (\alpha + \theta_1 + \theta_2) = n\pi \tag{2.8}$$

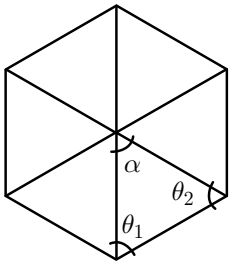


Figure 2.5: The central angles of an arbitrary triangle umbrella.

which can be rewritten as

$$\sum_{T=1}^{n}(\theta_1 + \theta_2) = n\pi - \sum \alpha_n \tag{2.9}$$

Note that the left hand side approximates the sum of the inscribed polygon interior angles. Substitution into Equation (2.7) gives

$$\int K da \approx \left(n\pi - \sum \alpha_n\right) + 2\pi - n\pi \tag{2.10}$$

$$\approx 2\pi - \sum \alpha_n \tag{2.11}$$

To convert the total (integral) curvature into a curvature value that will be assigned to the central vertex in the triangle umbrella, we need to divide both sides of Equation (2.11) by the area over which the integration takes place. Keeping in mind that a curvature value is ultimately assigned at every vertex, and that the overall total area over which we integrate should be equal to the total surface area, we need to somehow apportion the area of each triangle face in the umbrella to each of its three vertices.

The required apportioning of area has been done by other authors in various ways; for examples of different area operators used in this context, see (Meyer et al. 2003) and (Jagannathan and Miller 2007). Perhaps the simplest approach is to assign $1/3$ of the area of each face $\mathcal{A}$ to each of its vertices, and, of course, to sum the contributions from each face at common vertices. Dividing both sides of Equation (2.11) by this integration area gives

$$K \approx \frac{(2\pi - \sum \alpha_n)}{\sum \mathcal{A}(f_n)/3} = \frac{3(2\pi - \sum \alpha_n)}{\sum \mathcal{A}(f_n)} = \tilde{K} \tag{2.12}$$

This completes the derivation. □

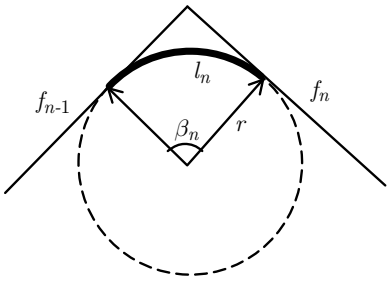


Figure 2.6: Edge smoothed with a cylinder segment.

Referring again to Figure 2.2, **mean curvature** at the point $\mathbf{p}_0$ is estimated by

$$\tilde{H}(\mathbf{p}_0) = \frac{3}{4}\left(\frac{\sum \beta_n \, ||e_n||}{\sum \mathcal{A}(f_n)}\right) \tag{2.13}$$

As was the case with the Gaussian curvature estimator, this estimator is also generally valid in the case of any adjacency point count greater than or equal to three.

This mean curvature estimator can be derived as follows:

We begin with the observations that, with any triangle mesh, the curvature of each (flat) face is zero, and the curvature along edges and vertices is undefined because the mesh is not smooth there. So, we consider the effects of smoothing each edge of the mesh using a small arbitrary radius cylinder placed such that mesh faces at the edge are tangent to the cylinder. Figure 2.6 shows an edge corner, in cross-section, smoothed by a circular arc (in bold) with length l. Note that, because of the tangential placement of the cylinder, the central angle associated with this smoothing arc is exactly equal to the previously identified dihedral angle β between adjacent face normals. By elementary geometry we have

$$l_n = \beta_n r \tag{2.14}$$

Now, since the mean curvature of a cylinder is everywhere $1/2r$ and the area of a smoothed edge is l times the length of the edge, the *integral (total) mean curvature* over a complete triangle umbrella is estimated by

$$\int H da \approx \sum_{k=1}^{n}\Big((1/2r)\, l_n \, ||e_n||\Big) = \sum_{k=1}^{n}\left(\frac{l_n \, ||e_n||}{2r}\right) \tag{2.15}$$

Substitution using Equation (2.14) gives

$$\int H da \approx \sum_{k=1}^{n} \left(\frac{\beta_n r \, ||e_n||}{2r} \right) = \sum_{k=1}^{n} \left(\frac{\beta_n \, ||e_n||}{2} \right) \tag{2.16}$$

Similarly to the case of Gaussian curvature, we need to divide both sides of Equation (2.16) by the area over which the integration takes place. As before, we use the simple $1/3$ area operator. However, with mean curvature, edge length also needs to be apportioned between vertices. Consistent with our area operator justification, we assign $1/2$ the length of every edge to its adjacent vertices, which results in

$$H \approx \sum_{k=1}^{n} \Big(\frac{\beta_n \left(||e_n||/2 \right)}{2} \Big) \Big/ \sum_{k=1}^{n} \mathcal{A}(f_n)/3 = \frac{3}{4} \left(\frac{\sum \beta_n \, ||e_n||}{\sum \mathcal{A}(f_n)} \right) = \tilde{H} \tag{2.17}$$

This completes the derivation. □

Generally, in practical applications, only the point data is given. So, for Gaussian curvature, the angles α_n and the face areas $\mathcal{A}(f_n)$ need to be calculated. Additionally, for mean curvature, the dihedral angles β_n and the edge lengths e_n need to be calculated. With reference to Figure 2.2, this can be done using vector operations as follows:

$$\alpha_n = \cos^{-1} \left(\frac{(\mathbf{p}_n - \mathbf{p}_0) \cdot (\mathbf{p}_{n+1} - \mathbf{p}_0)}{||(\mathbf{p}_n - \mathbf{p}_0)|| \; ||(\mathbf{p}_{n+1} - \mathbf{p}_0)||} \right) \tag{2.18}$$

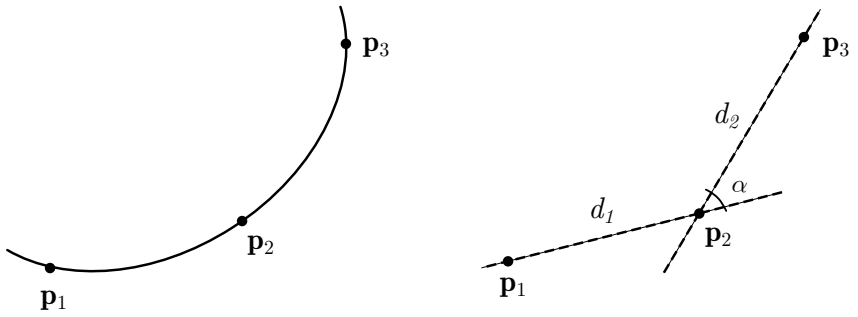
$$\mathcal{A}(f_n) = \frac{||(\mathbf{p}_n - \mathbf{p}_0) \times (\mathbf{p}_{n+1} - \mathbf{p}_0)||}{2} \tag{2.19}$$

$$\beta_n = \cos^{-1} \left(\frac{\big((\mathbf{p}_{n-1} - \mathbf{p}_0) \times (\mathbf{p}_n - \mathbf{p}_0)\big) \cdot \big((\mathbf{p}_n - \mathbf{p}_0) \times (\mathbf{p}_{n+1} - \mathbf{p}_0)\big)}{||(\mathbf{p}_{n-1} - \mathbf{p}_0) \times (\mathbf{p}_n - \mathbf{p}_0)|| \; ||(\mathbf{p}_n - \mathbf{p}_0) \times (\mathbf{p}_{n+1} - \mathbf{p}_0)||} \right) \tag{2.20}$$

$$e_n = ||(\mathbf{p}_n - \mathbf{p}_0)|| \tag{2.21}$$

2.2.2 Surface Cut

The development of the estimators presented in this section starts with curvature estimation for (discrete) digitized planar curves.



(a) (b)

Figure 2.7: Three surface points: (a) surface cross-section, (b) surface digitization.

Consider a planar curve contained in a surface, along with three points on the curve, as shown in Figure 2.7(a). Note that the unique plane determined by the three points qualifies as a cutting plane as defined in Section 1.3.1. Of course, for example, when working with a digitization that includes these three points, we can only estimate the position of the intermediate connecting points. One very straightforward way to estimate the location of the missing detail is to connect the points with line segments as illustrated in Figure 2.7(b).

Recall, from Section 1.2.2, that (signed) planar line curvature for smooth curves is defined as incremental change in angle divided by incremental change in length. This definition can be applied to a discrete digitization as follows:

With reference to Figure 2.7(b), the curvature at the point $\mathbf{p}_2$ is estimated by

$$\tilde{k}(\mathbf{p}_2) = \frac{\alpha}{(d_1 + d_2)/2} \tag{2.22}$$

where d_1 and d_2 are the distances from $\mathbf{p}_2$ to $\mathbf{p}_1$ and $\mathbf{p}_3$ respectively. Note that he total length $(d_1 + d_2)$ has been divided by two because some of this total will be apportioned to curvature estimation at the points $\mathbf{p}_1$ and $\mathbf{p}_3$. The reasoning behind this length operator is similar to the reasoning used for the face area and edge length operators used in Section 2.2.1.

To make use of the n-cut mean curvature theory from Section 1.3.1, we require digitizations in which equally spaced (by angle) cutting planes can

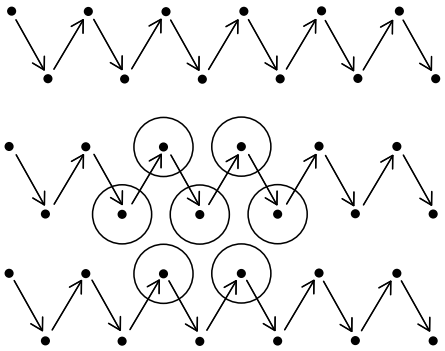
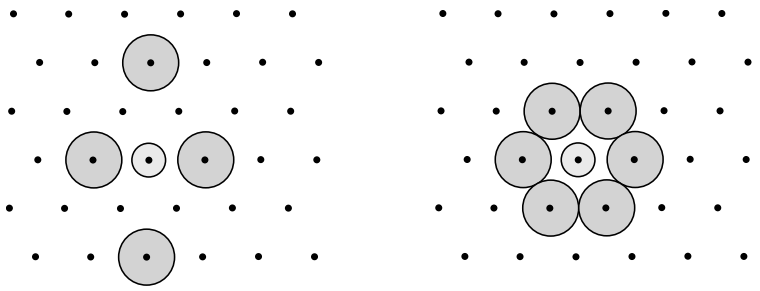


Figure 2.8: A regular acquisition pattern that results in hexagonal adjacency.



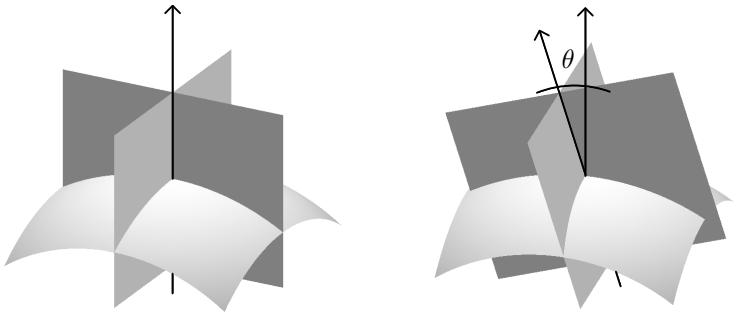
(a) (b)

Figure 2.9: Estimator point selection for: (a) two cut, (b) three-cut.

be found. Fortunately, surface digitizations are often acquired mechanically in a regular stepped fashion that results in a symmetry which can be exploited. A common example is illustrated in Figure 2.8, where the acquisition sequence results in a symmetrical hexagonal point adjacency pattern.

Given hexagonal adjacency, both two and three equally spaced cutting planes can be placed by selecting points as illustrated in Figure 2.9. The curvatures associated with these cutting planes can then be used to estimate surface curvature. For the two-cut mean curvature estimator, we have

$$\tilde{H}_{\text{(2-cut)}} = \frac{\tilde{\kappa}_1 + \tilde{\kappa}_2}{2} \tag{2.23}$$



(a) (b)

Figure 2.10: Surface cutting planes: (a) aligned with surface normal, (b) misaligned by angle θ.

and, for the three-cut mean curvature estimator,

$$\tilde{H}_{(3\text{-cut})} = \frac{\tilde{\kappa}_1 + \tilde{\kappa}_2 + \tilde{\kappa}_3}{3} \tag{2.24}$$

These estimators have been derived under the assumption that the intersection of the cutting planes aligns with the surface normal as shown, for example, in Figure 2.10(a). When the cutting planes do not align with the surface normal, a correction, based on Meusnier's theorem (see e.g. (Hermann and Klette 2007)), can be applied, resulting in the following general form for the n-cut estimator:

$$\tilde{H}_{(n\text{-cut})} = \cos\theta \left(\frac{\sum \tilde{\kappa}_n}{n} \right) \tag{2.25}$$

where θ is the misalignment angle as shown, for example, in Figure 2.10(b).

As was stated previously, in practical applications, only the point data is given. So, for the surface cut estimator, the angular advance α and the segment lengths d_n need to be calculated. With reference to Figure 2.7(b), this can be done using vector operations as follows:

$$\alpha = \cos^{-1}\left(\frac{(\mathbf{p}_2 - \mathbf{p}_1) \cdot (\mathbf{p}_3 - \mathbf{p}_2)}{||(\mathbf{p}_2 - \mathbf{p}_1)||\ ||(\mathbf{p}_3 - \mathbf{p}_2)||} \right) \tag{2.26}$$

$$d_n = ||(\mathbf{p}_{n+1} - \mathbf{p}_n)|| \tag{2.27}$$

Additionally, for misaligned cutting planes, θ needs to be calculated. We present this calculation in several steps, beginning with an estimation

of the surface normal vector as a weighted sum (by face area) of the adjacent the face normals. With reference to Figure 2.2, this sum is

$$\mathbf{v}_1 = \sum 2\,\mathcal{A}(f_n)\;\mathbf{n}_n \tag{2.28}$$

$$= \sum \Big(||(\mathbf{p}_n - \mathbf{p}_0) \times (\mathbf{p}_{n+1} - \mathbf{p}_0)||\;\big((\mathbf{p}_n - \mathbf{p}_0) \times (\mathbf{p}_{n+1} - \mathbf{p}_0)\big)\Big) \tag{2.29}$$

Note that, to minimize computation steps, the weighting used is double the face area. This is acceptable because normalization of $\mathbf{v}_1$ occurs later.

Next, working with any two of the cutting planes, we calculate a cutting plane intersection vector. With reference to Figure 2.7, and using additional subscripts a and b to distinguish between the two cutting planes, we calculate

$$\mathbf{v}_2 = \Big((\mathbf{p}_2 - \mathbf{p}_{a1}) \times (\mathbf{p}_{a3} - \mathbf{p}_2)\Big) \times \Big((\mathbf{p}_2 - \mathbf{p}_{b1}) \times (\mathbf{p}_{b3} - \mathbf{p}_2)\Big) \tag{2.30}$$

Finally, since what's required in Equation 2.25 is not θ, but rather $\cos\theta$, with reference to Figure 2.10, we calculate

$$\cos\theta = \frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{||\mathbf{v}_1||\;||\mathbf{v}_2||} \tag{2.31}$$

2.3 A Similarity Curvature Estimator

From Section 1.4.2, similarity curvature is, by definition, determined by the principal curvatures. But, because a digitization acquisition pattern will not, in general, align with the principal curvature directions, we cannot directly compute the principal curvatures. However, recall, from Section 2.2.1 and Section 2.2.2, that we do already have estimators for mean and Gaussian curvature.

Building on this, similarity curvature is estimated using the following process. First the mean and Gaussian curvatures are estimated, then the principal curvatures are calculated from the mean and Gaussian curvatures as follows:

$$\tilde{\kappa}_1 = \tilde{H} + \sqrt{\tilde{H}^2 - \tilde{K}} \tag{2.32}$$

$$\tilde{\kappa}_2 = \tilde{H} - \sqrt{\tilde{H}^2 - \tilde{K}} \tag{2.33}$$

Finally, the (estimated) similarity curvature is calculated from these principal curvatures using Definition 12 and Definition 13 as was given in Section 1.4.2.

Equations (2.32) and (2.33) are easily verified. Substitution using Equations (1.27) and (1.28) gives

$$\begin{aligned}
H + \sqrt{H^2 - K} &= \left(\frac{\kappa_1 + \kappa_2}{2}\right) + \sqrt{\left(\frac{\kappa_1 + \kappa_2}{2}\right)^2 - \kappa_1\kappa_2} && (2.34)\\
&= \left(\frac{\kappa_1 + \kappa_2}{2}\right) + \sqrt{\left(\frac{\kappa_1^2 + 2\kappa_1\kappa_2 + \kappa_2^2}{4}\right) - \left(\frac{4\kappa_1\kappa_2}{4}\right)} && (2.35)\\
&= \left(\frac{\kappa_1 + \kappa_2}{2}\right) + \sqrt{\left(\frac{\kappa_1^2 - 2\kappa_1\kappa_2 + \kappa_2^2}{4}\right)} && (2.36)\\
&= \left(\frac{\kappa_1 + \kappa_2}{2}\right) + \left(\frac{\kappa_1 - \kappa_2}{2}\right) && (2.37)\\
&= \frac{\kappa_1}{2} + \frac{\kappa_2}{2} + \frac{\kappa_1}{2} - \frac{\kappa_2}{2} = \kappa_1 && (2.38)
\end{aligned}$$

and

$$\begin{aligned}
H - \sqrt{H^2 - K} &= \left(\frac{\kappa_1 + \kappa_2}{2}\right) - \sqrt{\left(\frac{\kappa_1 + \kappa_2}{2}\right)^2 - \kappa_1\kappa_2} && (2.39)\\
&= \left(\frac{\kappa_1 + \kappa_2}{2}\right) - \sqrt{\left(\frac{\kappa_1^2 + 2\kappa_1\kappa_2 + \kappa_2^2}{4}\right) - \left(\frac{4\kappa_1\kappa_2}{4}\right)} && (2.40)\\
&= \left(\frac{\kappa_1 + \kappa_2}{2}\right) - \sqrt{\left(\frac{\kappa_1^2 - 2\kappa_1\kappa_2 + \kappa_2^2}{4}\right)} && (2.41)\\
&= \left(\frac{\kappa_1 + \kappa_2}{2}\right) - \left(\frac{\kappa_1 - \kappa_2}{2}\right) && (2.42)\\
&= \frac{\kappa_1}{2} + \frac{\kappa_2}{2} - \frac{\kappa_1}{2} + \frac{\kappa_2}{2} = \kappa_2 && (2.43)
\end{aligned}$$

Finally, to estimate similarity curvature, the principal curvature estimates $\tilde{\kappa}_1$ and $\tilde{\kappa}_2$ can be plugged directly into Equation (1.88).

2.4 Estimators Summary

The estimators presented in this chapter can be divided into four groups: 1) n-neighborhood triangle umbrella Gaussian,

2) n-neighborhood triangle umbrella mean,
3) n-cut mean, with or without compensation,
4) similarity curvature - using both the Gaussian and mean curvatures.

In practice, the application of these estimators will depend on identifying point neighborhoods and, for the n-cut mean estimator, identifying points which give equally spaced cutting planes. We note that the n-cut uncompensated mean estimator calculation has the least number of operations and would thus have the fastest computational execution time.

Bibliography

Agam, G. and Tang, X.: 2005a, Accurate principal directions estimation in discrete surfaces, *3DIM '05: Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling*, IEEE Computer Society, Washington, pp. 293–300.

Agam, G. and Tang, X.: 2005b, A sampling framework for accurate curvature estimation in discrete surfaces, *IEEE Transactions on Visualization and Computer Graphics* **11**(5), 573–583.

Alboul, L. and van Damme, R.: 1996, Polyhedral metrics in surface reconstruction, *in* G. Mullineux (ed.), *The Mathematics of Surfaces VI*, Clarendon Press, Oxford, pp. 171–200.

Allen, B., Curless, B. and Popović, Z.: 2002, Articulated body deformation from range scan data, *ACM Trans. Graph.* **21**(3), 612–619.

Bloomenthal, J.: 1985, Modeling the mighty maple, *SIGGRAPH Comput. Graph.* **19**(3), 305–311.

Cooper, O., Campbell, N. and Gibson, D.: 2003, Automated meshing of sparse 3d point clouds, *SIGGRAPH '03: ACM SIGGRAPH 2003 Sketches & Applications*, pp. 1–1.

Curless, B.: 2000, From range scans to 3d models, *SIGGRAPH Comput. Graph.* **33**(4), 38–41.

Curless, B. and Levoy, M.: 1996, A volumetric method for building complex models from range images, *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, New York, pp. 303–312.

Davies, A. and Samuels, P.: 1996, *An Introducion to Computational Geometry for Curves and Surfaces*, Oxford University Press, Oxford.

do Carmo, M. P.: 1976, *Differential Geometry of Curves and Surfaces*, Prentice-Hall, New Jersey.

Dyn, N., Hormann, K., Kim, S., and Levin, D.: 2000, Optimizing 3d triangulations using discrete curvature analysis, *Mathematical Methods for Curves and Surfaces*, OSLO.

Foley, J. D., van Dam, A., Feiner, S. K. and Hughes, J. F.: 2002, *Computer Graphics: Principles and Practice in C*, Addison-Wesley, Boston.

Fournier, M., Dischler, J.-M. and Bechmann, D.: 2006, 3d distance transform adaptive filtering for smoothing and denoising triangle meshes, *GRAPHITE '06: Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, pp. 407–416.

Gargantini, I., Schrack, G. and Kwok, A.: 1993, Reconstructing multishell solids from voxel-based contours, *SMA '93: Proceedings on the second ACM symposium on Solid modeling and applications*, pp. 463–464.

Hermann, S. and Klette, R.: 2007, A comparative study on 2d curvature estimators, *Computing: Theory and Applications*, IEEE Platinum Jubilee Conference of The Indian Statistical Institute, Kolkata, pp. 584–589.

Hill, F. S. and Kelley, S. M.: 2007, *Computer Graphics Using OpenGL*, Pearson Education, Upper Saddle River.

Hu, M. K.: 1962, Visual problem recognition by moment invariants, *IRE Trans. Inform. Theory* **8**, 179–187.

Ikeuchi, K., Nakazawa, A., Hasegawa, K. and Ohishi, T.: 2003, The great buddha project: Modeling cultural heritage for vr systems through observation, *ISMAR '03: Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, p. 7.

Jagannathan, A. and Miller, E. L.: 2007, Three-dimensional surface mesh segmentation using curvedness-based region growing approach, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**.

Jordan, C.: 1887, *Cours d'Analyse de l'Ecole Polytechnique*, Paris, France.

Kalogerakis, E., Simari, P., Nowrouzezahrai, D. and Singh, K.: 2007, Robust statistical estimation of curvature on discretized surfaces, *SGP '07: Proceedings of the fifth Eurographics symposium on Geometry processing*, pp. 13–22.

Klette, R. and Rosenfeld, A.: 2004, *Digital Geometry*, Morgan Kaufmann, San Francisco.

Kobbelt, L. P., Botsch, M., Schwanecke, U. and Seidel, H.-P.: 2001, Feature sensitive surface extraction from volume data, *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, New York, pp. 57–66.

Kovalevsky, V.: 2003, Multidimensional cell lists for investigating 3-manifolds, *Discreet Applied Mathematics* **125**, 25–44.

Kovalevsky, V.: 2008, *Geometry of locally Finite Spaces*, Editing House Dr. Baerbel Kovalevski, Berlin.

Lengyel, E.: 2004, *Mathematics for 3D Game Programming and Computer Graphics*, Charles River Media, Boston.

Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J. and Fulk, D.: 2000, The digital Michelangelo project: 3D scanning of large statues, *Proc. SIGGRAPH*, pp. 131–144.

Meek, D. S. and Walton, D. J.: 2000, On surface normal and gaussian curvature approximations given data sampled from a smooth surface, *Comput. Aided Geom. Des.* **17**(6), 521–543.

Meyer, M., Desbrun, M., Schrder, P., and Barr, A. H.: 2003, *Discrete differential-geometry operators for triangulated 2-manifolds*, Springer-Verlag, pp. 35–57.

Mitra, N. J. and Nguyen, A.: 2003, Estimating surface normals in noisy point cloud data, *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, pp. 322–328.

Mokhtarian, F. and Bober, M.: 2003, *Curvature Scale Space Representation: Theory, Applications, and MPEG-7 Sandardization*, Kluwer, Dordrecht.

Montani, C. and Scopigno, R.: 1990, Rendering volumetric data using sticks representation scheme, *SIGGRAPH Comput. Graph.* **24**(5), 87–93.

Oprea, J.: 2007, *Differential Geometry and its Applications*, Mathematical Association of America, Washington.

Pesic, P. (ed.): 2005, *Gauss, C. F.: General Investigations of Curved Surfaces (Reprint of publications from 1825 and 1827)*, Dover Publications, New York.

Rogers, D. F. and Satterfield, S. G.: 1980, B-spline surfaces for ship hull design, *SIGGRAPH '80: Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, pp. 211–217.

Rugis, J.: 2006, Extracting surface curvature from noisy scan data, *Image and Vision Computing New Zealand 2006*, pp. 185–189.

Rugis, J. and Klette, R.: 2006, Surface registration markers from range scan data, *LNCS: Combinatorial Image Analysis*, Vol. 4040/2006, Springer, Berlin / Heidelberg, pp. 430–444.

Sapiro, G.: 2001, *Geometric Partial Differential Equations and Image Analysis*, Cambridge University Press, Cambridge.

Tang, C., Medioni, G. and Lee, M.: 1999, Epipolar geometry estimation by tensor voting in 8d, *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.* **1**, 502–509.

Tang, X. and Agam, G.: 2004, Curve-based local surface geometry estimation in polyhedral surfaces, *Vision Geometry XII*, pp. 53–61.

Tong, W. and Tang, C.: 2005, Robust estimation of adaptive tensors of curvature by tensor voting, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**(3), 434–449.

Veblen, O.: 1905, Theory of plane curves in non-metrical analysis situs, *Trans. American Mathematical Society* **6**, 83–98.

Šrámek, M.: 1994, Fast surface rendering from raster data by voxel traversal using chessboard distance, *VIS '94: Proceedings of the conference on Visualization '94*, IEEE Computer Society Press, Los Alamitos, pp. 188–195.