A Theory of Specular Surface Geometry

Michael Oren
Department of Computer Science
Columbia University
New York, NY 10027
orenc@cs.columbia.edu

Abstract
A theoretical framework is introduced for the perception of specular surface geometry. When an observer moves in three-dimensional space, real scene features, such as surface markings, remain stationary with respect to the scene they belong to. In contrast, a virtual feature, which is the specular reflection of a real feature, travels on the surface. Based on the notion of causality, a novel feature classification algorithm is developed that distinguishes real and virtual features from their image trajectories that result from observer motion. Next, using support functions of curves, a closed-form relation is derived between the image trajectory of a virtual feature and the geometry of the specular surface it travels on. It is shown that in the 2D case where camera motion and the surface profile are planar, the profile is uniquely recovered by tracking just two unknown virtual features. Finally, these results are generalized to the case of arbitrary 3D surface profiles that are traveled by virtual features when camera motion is not confined to a plane. An algorithm is developed that uniquely recovers 3D surface profiles using a single virtual feature tracked from the occluding boundary of the object. All theoretical derivations and proposed algorithms are substantiated by experiments.

1 Specular Surfaces
This paper focuses on mirror-like reflection from smooth surfaces like glass, ceramic, polished metal, and plastic. Although the physics and geometry that govern specular reflection are well understood, visual interpretation of specular surfaces remains an open problem. Two major issues are associated with specular reflection. The first is detection of specularity. How can we determine whether an image feature corresponds to an actual scene point or whether it is the specular reflection of another scene point? This ambiguity poses a problem for all vision techniques that are based on feature detection and matching, such as, binocular stereo and structure from motion. At present, these techniques simply produce incorrect results when confronted with specular surfaces [Blake-1985],[Waldon and Dyer-1992]. The second problem, which is even more challenging, is shape recovery of specular surfaces. This problem is hard from an analytical perspective and currently only structured (active) illumination techniques can estimate the shape of a specular object [see Hirsch-1981], [Nayar et al.-1990],[Sanderson et al.-1988] and [Schultz-1994].

The ambiguity that specular surfaces introduce into image analysis arises from the existence of two distinctly different types of image features: real and virtual. A real feature corresponds to a physical scene point such as a surface marking or a surface texture element. On the other hand, a virtual feature is the reflection by a specular surface of another physical scene point that travels over the surface when the observer moves. Given the fundamental nature of the difference between real and virtual features, they must be distinguished before they are used (or discarded) by existing vision algorithms.

This classification problem is non-trivial since the photometric properties of a virtual feature could be identical to those of a real one. Consequently, brightness-based methods for identifying specular highlights (reflections of light sources), such as in [Ullman-1976],[Szymanski-1985],[Brebner and Binfrod-1987],[Brebbia and Blake-1988] and [Lee-1992], are limited in their applicability.

Since real and virtual features can be indistinguishable from a single image, we seek to investigate specular surfaces in the context of a moving observer. Two questions arise in this setting: First, how can real and virtual features be distinguished from their image trajectories? Second, what information regarding surface shape is contained in the image trajectory of a virtual feature? These questions have also intrigued other investigators in the past [Longuet-Higgins-1980], [Koenen and van Doorn-1988],[Blake-1985],[Blake and Brebbia-1984],[Blake and Binfrod-1989],[Waldon and Dyer-1993]. However, we still lack a complete understanding of what information regarding scene geometry can be extracted from specularities.

It turns out that even a moving observer, exploitation of virtual features is a difficult problem. Zisserman et al. [Zisserman et al.-1989] showed that a moving observer can determine a surface profile by tracking the reflection of a known light source, but only up to a one-parameter family of curves. In other words, even with a known source, shape cannot be uniquely recovered. While their result shows the existence of a family of curves, it does not provide a closed-form expression for this family. It was only recently that Ballez-Chehrebi...
2.1 Curve Representation

First, we present a general representation of curves which is based on the support function. This representation will play a crucial role in the classification of real and virtual features as well as the recovery of specular surfaces.

2.1.1 Envelopes and the Legendre Transform

The most common description of a 2D curve is its representation as a collection of points given by their Cartesian coordinates, \((x, y)\) (see Figure 1(a)). Though this representation is conceptually simple, it results in complex equations when used to describe the geometry of specular reflection. The direction of the reflected ray changes with observer motion and hence there is no convenient Cartesian coordinate system that can be used to express the constraints on the slope and position of the reflecting point. A more natural choice is to parameterize the curve by its slope. The curve may then be viewed as an envelope of surface tangents rather than a collection of points, as illustrated in Figure 1(b).

What is the most suitable representation for an envelope of tangents? Beliveau-Ceballos and Rodriguez-Danta (1992) suggested using the Legendre transform of the curve equation. The Legendre transform, \(\Psi(r)\), of a differentiable function of one variable, \(\psi(r)\), is:

\[
\Psi(r) = \frac{\mathcal{E} - r^2}{2} \quad \text{where} \quad \mathcal{E} = \psi(x) (1)
\]

A geometric interpretation of the Legendre transform is illustrated in Figure 2(a). If \((x, \mathcal{E})\) is the Cartesian representation of a curve, then \(\mathcal{E}\) is the tangent of the angle between the tangent line and the \(y\)-axis, and \(\mathcal{E}\) is the intersection of the tangent line with the \(x\)-axis.

2.1.2 Support Function of a Curve

While the Legendre transform of a curve simplifies the treatment of specular reflectance (see Beliveau-Ceballos and Rodriguez-Danta, 1992), it suffers from a few drawbacks. A major disadvantage is that neither \(\Psi\) nor \(\mathcal{E}\) undergo simple transformations under rotation of the coordinate system, a property that is highly desirable in our work for reasons that will become clear in due course. This motivated us to represent the tangent line not with the slope and intersection point but with its...
Figure 2: (a) Representation of a tangent using $(\psi, k)$; (b) Representation of a tangent using the support function, i.e. the distance $p_s$ from the origin and the normal angle $\theta_s$.

The function, $p_s(\theta_s)$, is called the support function of the curve (Guggenheim-1977).

Given the Cartesian representation $(x, y)$ of a curve, the support function is simply:

$$\theta_s = \arctan \left( \frac{y}{x} \right)$$

$$p_s = r_s \sin \theta_s = \rho \cos \theta_s$$

Calculating the derivative of $p_s$ in (3) with respect to $\theta_s$ and using (2) yields the following mapping:

$$\begin{pmatrix} p_s \\ \rho_s \end{pmatrix} = \begin{pmatrix} \cos \theta_s & sin \theta_s \\ -\sin \theta_s & \cos \theta_s \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

From the above relation, the inverse transform is easily determined as:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos \theta_s & -\sin \theta_s \\ \sin \theta_s & \cos \theta_s \end{pmatrix} \begin{pmatrix} p_s \\ \rho_s \end{pmatrix}$$

This representation has the advantage that it depends explicitly on the slope of the curve. Further, under a rotation of the coordinate system, $p_s$ remains unchanged while $\theta_s$ is only subjected to a simple linear shift.

2.2 2D Caustics and Feature Classification

We are now ready to address the classification of features into real and virtual ones. Figure 7 shows a single image taken from a sequence obtained by a moving stereo. Two image features (1 and 2) are highlighted. Feature 1 is a real surface marking (a checkerboard corner point on the surface of the sphere) while feature 2 is the reflection of a scene feature (also a checkerboard corner). Despite these differences, the two features appear almost identical and hence are indistinguishable from a single image. We introduce an algorithm that allows a moving observer to quickly discriminate between real and virtual features without making any assumption regarding the photometric properties of the features.

When the sensor moves around the object, the virtual feature travels on the specular surface producing a family of reflected rays. The envelope defined by this family is called a caustic. The caustic of a real feature is nothing but a point, the actual position of the feature in the scene where all the reflected rays intersect. But for a virtual feature the caustic will be a curve. Therefore, to classify a feature, all we need to compute is the caustic and test whether it is a point or a curve.

Pertinent information that the image of a virtual feature contains is the direction, $\theta_s$, of the reflected ray relative to the world coordinate system and its signed distance, $p_s$, from the origin of the coordinate system. These parameters can be computed in a straightforward manner from the position of the feature in the image and the camera parameters (position, orientation, and focal length).

The caustic that we seek to compute is tangent to each of the reflected rays. Hence, $\rho_s(\theta_s)$ (more precisely, $\rho_s(\theta_s + \frac{\pi}{2})$) represents the support function of the caustic. Therefore, we will use the term caustic and image-trajectory of a feature interchangeably since they convey the same information. Note that we have used $\rho_s$ for the support function of the caustic, to distinguish it from the support function of the specular surface profile which will be denoted by $\rho_s$.

The computation of the caustic curve $\{x, y\}$ (parameterized by $\theta_s$) in Cartesian coordinates is straightforward. Given $\rho_s(\theta_s)$, expression (3) can be used to get:

$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{pmatrix} \cos \theta_s & -\sin \theta_s \\ \sin \theta_s & \cos \theta_s \end{pmatrix} \begin{pmatrix} p_s \\ \rho_s \end{pmatrix}$$

where $\theta_s$ is the derivative of $\rho_s$ with respect to $\theta_s$. As stated earlier, the conspicuousness of the above caustic gives us a direct means of classifying real and virtual features. At times, however, virtual features may produce a compact caustic. Specifically, when the radius of curvature of the profile is very small, for example in a sharp corner (Koenderink and van Doorn 1984), the caustic will be compact making it hard in the presence of noise to determine that the feature is virtual one. However, such virtual features are almost fixed in space and behave like real feature points. As a result, they can be treated as real features and effectively used in techniques such as stereo and motion.

---

2This assumption is valid in the case of a parabolic specular surface with a scene feature exactly on its axis. It is easy to convince oneself of the validity of these recovery equations to be derived in section 3.4. This is clearly a special case that is unlikely from a practical perspective.
2.3 Experiments: Feature Classification

To illustrate classification we used the metallic sphere shown in Figure 3. The real and virtual features shown in the images were tracked while the sphere was moved around in a planar trajectory. The two features are tracked from one frame to the next using the sum of square difference (SSD) correlation operator. The computed support functions \( \rho_1(\theta) \) of the two features are shown in Figure 4(a). From each support function a caustic curve was computed using expression (6).

In Figure 4(b), the two caustics are plotted as parameterized curves in the \( x-y \) plane. As expected, the caustics of the real feature is a small cluster centered around the actual position of the feature. In contrast, the caustic of the virtual feature is a curve with a cusp (which is typical of caustics resulting from specular reflection). Several tests can be employed to determine the contrast of caustics. We found the second moment of the caustic computed with respect to the centroid to be a simple but effective measure. In the above experiment, second moments of the real and virtual caustics were found to have a ratio of 30:1, clearly sufficient for reliable classification.

![Figure 4](image)

Figure 4: (a) Support functions produced by the two features shown in Figure 3. (b) Caustics computed from the support functions and plotted on the \( x-y \) plane. The caustic of the real feature is a compact cluster, while that of the virtual feature is a curve with a cusp.

2.4 2D Profile Recovery

We now examine the problem of recovering 2D specular profiles by moving the sensor and tracking virtual features. In this section, we derive new expressions that relate surface profile to image trajectory using the support function representation. We also show how tracking of two or more unknown virtual features, enables us not only to find the position of the corresponding some features (or sources), but also to recover the profile of the specular surface without any ambiguity. This is shown to be possible without prior knowledge of any profile point.

2.4.1 The Profile Equation

Figure 5 shows a specular surface profile and some features reflected by it in the direction of the camera. The camera is moved in the plane of the profile and each feature is tracked in image space. We assume that all scene features are relatively far, so that any given feature's direction is the same for all points on the surface profile, i.e., for each feature, \( \beta \) is nearly constant over the entire profile. As in section 2.2, we use the notation \( \rho_1(\theta) \) for the profile support function and \( \rho_2(\theta) \) for the caustic support function. By \((x, y)\) we denote the Cartesian coordinates of the reflecting point on the profile. The reflecting point lies on two tangents. The first is a tangent to the profile curve whose normal is \( \beta_1 \). From (4) we get:

\[
\rho_1(\theta_1) = x \cos \theta_1 + y \sin \theta_1
\]

The second tangent is the reflected ray which is a tangent to the caustic. The normal to the caustic at the tangential point is \( \beta_2 + \frac{\pi}{2} \) (see Figure 5). Again, using (4) we have:

\[
\rho_2(\theta_2) = x \cos (\theta_2 + \frac{\pi}{2}) + y \sin (\theta_2 + \frac{\pi}{2})
\]

From these two equations we get the following expression for the \( z \)-coordinate of the profile point:

\[
x = \frac{\cos \theta_1 \rho_1(\theta_1) - \sin \theta_2 \rho_2(\theta_2)}{\cos(\theta_1 - \theta_2)}
\]

Also, from expression (5) we get:

\[
x = \cos \theta_1 \rho_1(\theta_1) - \sin \theta_2 \rho_2(\theta_2)
\]

Equating the above two expressions for \( x \), and using the law of specular reflection:

\[
\beta_1 = \beta_2 + \frac{\pi}{2}
\]

we get:

\[
\rho_2(\beta_2 - \beta_1) = -\sin(\beta_2 - \beta_1) \rho_1 + \cos(\beta_2 - \beta_1) \rho_2
\]

or:

\[
\rho_2(\beta_2 - \beta_1) = \frac{d}{d\theta_2} \left[ \cos(\theta_2 - \theta_1) \rho_1(\theta_2) \right]
\]

This differential equation is fundamental to our analysis as it relates the support function \( \rho_1 \) of a feature to the surface profile \( \rho_2 \) that we seek to recover. We need to integrate this equation to retrieve the desired profile support function \( \rho_1(\theta_1) \). Since in practice we work with the angle of reflectance \( \beta_1 \) and not with \( \theta_1 \) (the unknown normal direction) we substitute equation (9) into (11) and

\[
\rho_2(\beta_2 - \beta_1) = \frac{d}{d\theta_2} \left[ \cos(\theta_2 - \theta_1) \rho_1(\theta_2) \right]
\]
integrate over \( \delta \):

\[
\rho_{n}^{a} = \frac{\int_{0}^{\frac{\pi}{2}} \rho_{a}^{b} \rho_{a}^{c} \cos^2 \left( \frac{\delta}{2} \right) d\delta}{\cos \left( \frac{\delta}{2} \right)}
\]

(12)

where, \( \delta \) is the starting angle of the integration. The last expression gives the support function of the unknown profile as an integral of the support function of the caustic which is measured by the moving observer. The Caustic coordinates of the surface profile can be computed using equation (5), or alternatively, using equation (21) which was derived in section 7 and does not require the computation of the derivative of \( \rho_{a} \). An important observation from equation (12) is that even if the feature direction is given, the surface profile cannot be determined completely due to the unknown constant of integration: \( C = \rho_{a}^{b} \cos^2 \left( \frac{\delta}{2} \right) \). This implies that the surface profile is determined only up to a one-parameter family of curves. In the next section, we will show how this ambiguity can be resolved.

2.4.2 Recovery of a 2D-Profile Using Multiple Features

As we saw in the previous section, even if we track a feature whose position is known, the surface profile cannot be recovered uniquely. Consider two scene features in the directions \( a_{k} \) \((k = 1, 2)\). We do not assume these angles are known, or that any point on the surface is given a priori. By moving the camera around the profile in a known trajectory and tracking two features we get two caustics \( \rho_{a_{k}}^{b} \) \((k = 1, 2)\). Each caustic determines, based on equations (12), a two parameter family of profile support functions, \( \rho_{a_{k}}^{b} \) \((k = 1, 2)\). Each support function depends on two parameters, the feature direction and the constant of integration: \( C = \rho_{a_{k}}^{b} \cos^2 \left( \frac{\delta}{2} \right) \). The two parameters \( a_{k} \) will be reflected on the profile whose normals lie in the range \( \delta - 90^\circ, \delta + 90^\circ \). Therefore, when the two features are located exactly at diametrically opposite sides of the object, i.e. \( 180^\circ \) apart, their profiles must overlap. In the overlap region, the two recovered support functions \( \rho_{a_{k}}^{b} \) must agree, as they represent the same profile segment. In the recovery process, we search the \( \rho_{a_{k}}^{b} \) parameter space and find the 4 parameters that minimize the distance between \( \rho_{a_{k}}^{b} \) and \( \rho_{a_{k}}^{b} \) in the overlapping region. Once the 4 parameters are found, the surface profile is reconstructed over the entire image of measurements, i.e. not just the overlap region, but all points on the profile traveled by either of the two features. The above described approach is easily extended to larger numbers of features to obtain a larger profile.

2.5 Experiments: Recovery of 2D Profile Using 2 Features

We conducted experiments on a variety of objects. Here, we present results on just two profiles, one circular and the other elliptical. The experimental setup used is shown in Figure 8(a). A specular object is positioned in a robot's workspace. A background

Figure 6: (a) The experimental setup. A camera is mounted on the end-effector of a 3 DOF robot. (b) A typical image including a large number of virtual features.

at a large distance from the object produces virtual features on the specular surface of interest. A camera attached to the end-effector of the robot is moved around the object. Two features were selected in the initial image (Figure 6(b)) and tracked through the image sequence using the SSD correlation operator. From the feature trajectories, the camera coordinates and focal length, support functions for each of the two features were computed (see Figure 7(a)). Next, the two unknown pairs of parameters \( \theta_{1}, C_{1} \) and \( \theta_{2}, C_{2} \) that minimize the distance between the two support functions in the overlap region were found via search. We used the mean-squared-distance metric to formulate the search. Once the four parameters are determined, the surface profile corresponding to each feature trajectory is independently recovered and then the two profiles are fused together to obtain a larger reconstructed profile. Figure 7(b) shows the recovered profile of the sphere. Experimental results for an oval-shaped object are shown in Figure 8. In both experiments we see that the specular profiles are estimated with high accuracy.

Figure 7: 2D profile of a sphere recovered by tracking two unknown features. (a) Support functions for the two features. (b) The recovered surface profile (solid line) and the actual profile (thin line). The directions of the two unknown features were found to be \( \theta_{1} = -28^\circ \) and \( \theta_{2} = 34^\circ \).

3 Recovery of 3D Surface Curves

We now generalise the results of the previous sections to 3D surfaces. The caustic motion is no longer confined to a plane, and consequently, the surface profile can be any smooth space curve. Figure 9 shows the tracking
of a virtual feature as it travels on a 3D curve. If we project the curve onto a 2D plane, for instance the 3-
axis plane, the projection of any surface normal on the
curve is not necessarily the bisector of the projections
of the incidence and reflection angles. In general, there
is no projection plane that obviates this problem.
In short, the problem of 3D curves cannot be reduced to
a finite number of 2D profile problems. However, our
results on 2D profiles have given us the basic tools and
understanding necessary to generalize our results to the
3D case.

![Figure 8: 2D profile of an ellipsoid obtained by tracking two unknown features. (a) Support functions of the two features. (b) The recovered surface profile (thick line) and the actual profile (thin line). The directions of the two unknown features were found to be $\theta = -26^\circ$ and $\theta = 50^\circ$.](image)

![Figure 9: Recovery of surface geometry for general caustic motion (along a 3D trajectory) cannot be decomposed into two 2D profile reconstruction problems.](image)

### 3.1 3D Caustics and Feature Classification

Analysis of 3D specular surfaces requires us to use
vectors for representations of incidence, reflection, and
normal directions. For general camera motion, the famil-
y of reflected rays, $s(t)$, is no longer confined to a
plane. Hence, the caustic corresponding to the reflected
rays is an arbitrary surface curve. We have parameterized
the family of reflected rays by $t$, which may be viewed
as a time parameter. The caustic curve is then given by
$\mathbf{x}(t)$. It is also parametrized by $t$ such that point $\mathbf{x}(t)$
on the caustic is tangent to the ray $\mathbf{s}(t)$. We use the
notations $s$ and $\mathbf{s}(t)$ for the feature direction vector and
the surface normal vector, respectively. All vectors de-
finite in the space $\mathbf{v}(t)$, $\mathbf{u}(t)$, $\mathbf{h}(t)$, and $\mathbf{n}(t)$, represent
only directions and are assumed to be unit vectors.

The main idea behind the derivation of the 3D caustic
curve is to decompose the caustic point position at
any given instant $t$ into two orthogonal components as
follows:

$$\mathbf{x}(t) = \tilde{\mathbf{L}}(t) + \left[\mathbf{x}(t), \mathbf{v}(t)\right] \mathbf{w}(t)$$

The first component, $\mathbf{L}(t)$, is the distance vector from
the ray $\mathbf{s}(t)$ to the origin $\mathbf{O}$ of the world coordinate
system. This vector is analogous to the support func-
tion, $\rho_0$, introduced in section 2.1.2. The vectors $\mathbf{v}(t)$ and $\mathbf{w}(t)$ are computed directly from the image trajec-
tory of a virtual feature and the known camera parame-
ters. Equation (13) expresses the unknown caustic curve
$\mathbf{x}(t)$ in a moving coordinate system that is attached to
the ray $\mathbf{s}(t)$. To determine $\mathbf{x}(t)$ we need to find the
unknown component $\left[\mathbf{x}(t), \mathbf{v}(t)\right] \mathbf{w}(t)$ along $\mathbf{s}(t)$.

The first step in the caustic curve detection is to dif-
ferentiate (13) with respect to $t$. This yields the tangent
to the caustic curve:

$$\mathbf{v}(t) = \tilde{\mathbf{L}}(t) + \left[\mathbf{x}(t), \mathbf{v}(t)\right] \mathbf{w}(t) + \left[\mathbf{x}(t), \mathbf{w}(t)\right] \mathbf{v}(t)$$

Taking the inner product of both sides of the above expression with $\mathbf{w}(t)$ and using $\left[\mathbf{w}(t), \mathbf{w}(t)\right] = 0$ we get:

$$\left[\mathbf{x}(t), \mathbf{v}(t)\right] = \left[\mathbf{L}(t), \mathbf{w}(t)\right] + \left[\mathbf{x}(t), \mathbf{w}(t)\right] \mathbf{v}(t)$$

$$\mathbf{v}(t) = \tilde{\mathbf{L}}(t) + \left[\mathbf{x}(t), \mathbf{w}(t)\right] \mathbf{v}(t)$$

Since $\mathbf{x}(t)$ is the caustic curve, for any given $t$ its tan-
gent $\mathbf{v}(t)$ is parallel to $\mathbf{s}(t)$: $\left[\mathbf{x}(t), \mathbf{s}(t)\right] = 0$. If we
plug this constraint in (15), we get an expression for
$\left[\mathbf{x}(t), \mathbf{w}(t)\right]$ that can be substituted back in (13). The
result is the 3D caustic curve equation:

$$\mathbf{x}(t) = \tilde{\mathbf{L}}(t) + \left[\tilde{\mathbf{L}}(t), \mathbf{w}(t)\right] \mathbf{v}(t)$$

Comparing this to (6) we see that the 2D caustic is a
special case of the 3D caustic, corresponding to planar
motion where the curve parameter is $t = 0$.

### 3.2 Experiments: 3D Caustic Curves and Feature Classification

![Figure 10: (a) Image trajectories, $\tilde{\mathbf{L}}(t)$, of two features plotted as parameterized curves. (b) Their computed 3D caustic curves.](image)

We conducted feature classification experiments on
the object in Figure 3, which produces both real and
virtual features. The camera motion is not planar but
rather an arbitrary smooth 3D trajectory. The caustics
shown in Figure 10(b) are computed from the image trajec-
tories shown in Figure 10(a) using equation (16). As
expected, the caustic curve of the real feature (1)
is a small cluster of points centered around the actual
feature position while the caustic curve of the virtual

745
feature (5) is a space curve. The ratio of the second moments of the two caustics was found to be 12.1.

### 3.3 Surface Curve Equation for 3D Camera Motion

As the observer moves around a spatial object, a virtual feature travels along a 3D profile on the object's surface. In this section, we show how this surface curve, \( \mathbf{x}(t) \), can be recovered from the image trajectory of the virtual feature given by the pair \( (\mathbf{L}(t), \mathbf{u}(t)) \). As in section 3.1, the key step is to decompose the surface curve into two orthogonal components as:

\[
\mathbf{z}(t) = \mathbf{L}(t) + \langle \mathbf{z}(t), \mathbf{u}(t) \rangle \mathbf{u}(t)
\]

(17)

The unknown quantity is the component of \( \mathbf{z}(t) \) along the reflected ray which is given by \( \langle \mathbf{z}(t), \mathbf{u}(t) \rangle \). To recover this component, we examine the support function of the surface curve which is the distance of the tangent plane from the origin \( \mathbf{O} \):

\[
\rho_0(t) = \langle \mathbf{z}(t), \mathbf{n}(t) \rangle
\]

(18)

where, \( \mathbf{n}(t) \) is the surface normal at \( \mathbf{z}(t) \). Differentiating the above expression with respect to \( t \) and using the fact that the vector \( \mathbf{z} \) is tangent to the surface gives:

\[
\dot{\rho}_0(t) = \langle \dot{\mathbf{z}}(t), \mathbf{n}(t) \rangle
\]

(19)

Now, the law of specular reflection can be written in vectorial form as:

\[
\dot{\mathbf{v}} = 2\langle \mathbf{v}, \mathbf{n} \rangle \mathbf{n} - \mathbf{v}
\]

(20)

Using this expression for \( \dot{\mathbf{v}} \) in (17) and substituting \( \mathbf{z}(t) \) back in (18) and (19) we get:

\[
\rho_0(t) = \langle \mathbf{z}(t), \mathbf{v}(t) \rangle \mathbf{n}(t) + \langle \mathbf{L}(t), \mathbf{n}(t) \rangle
\]

\[
\dot{\rho}_0(t) = \dot{\langle \mathbf{z}(t), \mathbf{v}(t) \rangle} \mathbf{n}(t) + \langle \dot{\mathbf{L}}(t), \mathbf{n}(t) \rangle
\]

Multiplying the last identity by \( \langle \mathbf{v}, \mathbf{n} \rangle \), the second by \( \langle \mathbf{z}, \mathbf{n} \rangle \), and adding the results we have:

\[
\rho_0(t) \langle \mathbf{z}, \mathbf{n}(t) \rangle + \langle \mathbf{z}(t), \mathbf{v}(t) \rangle \mathbf{n}(t) = \mathbf{L}(t) + \dot{\langle \mathbf{z}(t), \mathbf{v}(t) \rangle} \mathbf{n}(t)
\]

This result can be further simplified. In this end, we find the derivative of \( \rho \) with respect to \( t \) using (20). Since the feature direction \( \mathbf{v} \) is a constant we get:

\[
\frac{1}{2} \dot{\mathbf{z}}(t) = \langle \mathbf{z}, \mathbf{n}(t) \rangle \mathbf{n}(t) + \langle \mathbf{z}(t), \mathbf{n}(t) \rangle \mathbf{n}(t)
\]

(21)

Note that the right hand side of the above expression figures explicitly in (21). Substituting the left hand side instead gives the following fundamental relationship:

\[
\frac{d}{dt} \left( \rho_0(t) \langle \mathbf{z}, \mathbf{n}(t) \rangle \right) = \frac{1}{2} \langle \dot{\mathbf{z}}(t), \mathbf{n}(t) \rangle
\]

(22)

Then, by integration we have:

\[
\rho_0(t) = \frac{1}{2} \int \mathbf{L}(t') \langle \mathbf{z}(t'), \mathbf{n}(t') \rangle dt' + \rho_0(t_0) \langle \mathbf{z}(t_0), \mathbf{n}(t_0) \rangle
\]

(23)

Note that even for a known feature direction the above solution is determined only up to an unknown parameter, namely, the constant of integration: \( C = \rho_0(t_0) \langle \mathbf{z}(t_0), \mathbf{n}(t_0) \rangle \). The above ambiguity is inherent to the recovery problem; \( C \) is determined by \( \rho_0(t_0) \) which it is itself unknown. However, if the integration in (24) is started from an emptying boundary, the ambiguity is eliminated; the unknown constant vanishes since \( \langle \mathbf{z}(t_0), \mathbf{n}(t_0) \rangle = 0 \).

### 3.4 Curve Equation from Support Function

Our final result is a closed-form solution that allows the unique recovery of a surface curve from the above support function \( \rho_0(t) \). The unknown quantity we seek is \( \langle \mathbf{z}(t), \mathbf{n}(t) \rangle \). The inner product of (17) with the normal vector \( \mathbf{n}(t) \) can be written as:

\[
\langle \mathbf{z}(t), \mathbf{n}(t) \rangle = \langle \mathbf{z}(t), \mathbf{v}(t) \rangle \langle \mathbf{v}(t), \mathbf{n}(t) \rangle + \langle \mathbf{L}(t), \mathbf{n}(t) \rangle
\]

(25)

or:

\[
\langle \mathbf{z}(t), \mathbf{n}(t) \rangle = \langle \mathbf{L}(t) - \langle \mathbf{z}(t), \mathbf{n}(t) \rangle \mathbf{v}(t), \mathbf{n}(t) \rangle
\]

(26)

Substituting this back in (27) gives us the surface curve:

\[
\langle \mathbf{z}(t), \mathbf{n}(t) \rangle = \langle \mathbf{L}(t) + \dot{\langle \mathbf{z}(t), \mathbf{v}(t) \rangle} \mathbf{n}(t), \mathbf{n}(t) \rangle
\]

(27)

### 3.5 Experiments: Recovery of 3D Curves

![Figure 11](attachment:image.png)

Figure 11: (a) Actual trajectory of camera motion plotted on a sphere. (b) The tracking of a virtual feature. The trajectory is parameterized by time \( t \) shown here as the third dimension. The other two coordinates represent the distance vector \( \mathbf{z}(t) \).

Our last set of experiments are on recovery of 3D curves of spatial objects. The recovery of a 3D profile is a much harder problem than that of 2D profiles since the surface trajectories traveled by different features are not guaranteed to overlap over large surface curve segments. This forces us to use one feature at a time but resolve shape ambiguity by tracking each feature from the occluding boundary of the surface as explained in the section 3.3.

In these experiments we tracked the reflection of a point light source (highlighted) rather than the reflection of a scene feature. The two however are equivalent from a theoretical perspective. The camera direction trajectory used is plotted in Figure 11. The surface curve was recovered from the image trajectory of the virtual feature using equation (24). The recovered (dotted) and actual (solid) surface profiles are displayed in Figure 12 using two different viewpoints. We see that the recovered curve is in strong agreement with the actual surface curve.
4 Summary

In this paper, we explored visual information regarding specular surface geometry available to a moving observer. Specular reflection is a phenomenon that is ubiquitous in the real world; it is exhibited to some degree by most real surfaces. A sound understanding of specular surfaces and their appearance in brightness images is fundamental to progress in computational vision.

In this work, we have introduced a comprehensive mathematical framework for analyzing the relation between specular surface geometry and image trajectories of scene features reflected by the surface. It was shown that analyses of specular surfaces are tractable only if representations for surface curves and reflected rays are carefully chosen. We invoked support functions and the notion of caustic curves to represent trajectories produced by image features. Caustics were shown to hold valuable information regarding scene geometry.

Image features were categorized into two basic classes: real and virtual. Real features are the only sort that should be directly used by vision algorithms such as structure from motion. In contrast, virtual features are reflections of scene features by a specular surface. Unlike real features, they travel on specular surfaces when the observer changes his/her viewpoint. We showed that the curves of real and virtual features have distinctly different anatomies: one is a compact cluster while the other is an arbitrary space curve. These properties of feature caustics were used to develop a classification algorithm. This algorithm can serve as a useful precursor to a variety of vision techniques.

Finally, we showed that virtual features must not be quickly discarded as they contain valuable information regarding the shapes of specular objects in the scene. In the case of pure specular surfaces (smooth metals, glass, etc.), virtual features are the only available source of visual information. We derived shape recovery equations that relate the image trajectory of a virtual feature to the profile of the reflecting surface. Though the problem of specular profile recovery was considered severely under-constrained in the past, we demonstrated that it is feasible and can be performed with reasonable accuracy.

References


