J. Indian Inst. Sci., Nov.–Dec. 2006, **86**, 599–615

Shape from shading: Analysis by synthesis of implicit and general surfaces

K. RAJGOPAL* AND SRIRAM J. SATHISH
Department of Electrical Engineering, Indian Institute of Science, Bangalore 560 012, India.
e-mail: kasi@ee.iisc.ernet.in; Tel: 91-80-2293 2366 and Fax: 91-80-2360 0444.

Received on July 14, 2006; Revised on November 13, 2006.

Abstract

The shape from shading (SfS) problem in computer vision requires complete knowledge of the image conditions under which an image is created to produce surface descriptions. Almost all the methods rely on modeling the image formation process view and invert it mathematically which in turn places constraints on imaging conditions. These methods make constraining assumptions on camera model (orthographic projections), light source (single-point source at infinity) and reflectance model (Lambertian). In this paper, we present a general framework for solving the SfS problem under general imaging conditions by using powerful image synthesis techniques from computer graphics and thus moving the complexity of producing surface descriptions from analysis to the synthesis side. The technique relies on iterative synthesis of images from object descriptions in order to minimize an error function. The technique is illustrated in detail for quadric surfaces, with the ellipsoid as the specific example. To extend the method to general surfaces, the surfaces are modeled under the Bézier framework. The proposed framework is found to be very general with the capability to accommodate widely varying reflectance models and light source types.

Keywords: Computer vision, Analysis by synthesis, Bézier surfaces.

1. Introduction

Shape from shading (SfS) is one of the earliest problems in computer vision [1]. The problem can be stated as follows: Given the image of an object, extract the 3D information of the object, the object's surface profile by analyzing the shading on the image. The term *shading* is used to denote the variations in the intensity values of the object's image occurring due to variations in the object's surface profile. The recovered 3D shape is expressed in one of the many ways like (i) the relative depth $z(x, y)$ of the surface points with respect to a reference x–y plane, (ii) the normals about the various points on the surface, (iii) the surface gradient, representing the rate of change in depth along the x and y directions, etc.

Traditionally, the SfS problem is approached by first modeling the image formation process. The resulting image irradiance equation is a partial differential equation, which is then either explicitly solved [1] or used in a constrained optimization process [2] to obtain shape information. These methods solve the problem under very restrictive constraints and assumptions: the surface reflection process is typically Lambertian; the light is usually a

*Author for correspondence.

single-point source at infinity, and in one specific case [3] a uniform hemispheric source; the camera is almost always orthographic, and in rare instances perspective [4]. There have been very few attempts to solve the general SfS problem, even for a simple surface like the sphere. The SfS problem under general imaging conditions is solved in [5] where shape information at a known point on the surface is propagated across the entire surface to get the surface shape. To constrain the surface propagation method to give a unique solution, the author makes use of multiple images of the same scene.

In this paper, we attempt to address the SfS problem, first for a specific class of surfaces called implicit surfaces, and then for general surfaces, given auxiliary information like the type and position of the object, the number, type and distribution of light sources, the model for light reflectance on the surface along with the associated parameter values, etc. Image synthesis techniques from computer graphics are used for estimating an object's shape from its image.

In Section 2, a description of implicit surfaces with specific emphasis on their analysis is presented. Quadric surfaces are an important class of implicit surfaces. In Section 3, we present the analysis by synthesis framework and illustrate it for quadric surfaces. This framework is extended in Section 4 for general surfaces using Bézier techniques to model the surface as linear combinations of Bernstein polynomials. Finally, conclusions are presented in Section 5.

2. Analysis of implicit surfaces

Implicit surfaces are one of the most commonly occurring surfaces in computer graphics. An implicit surface in 3D space is defined by a real-valued function $\boldsymbol{F}$ over the variables x, y and z. With the function $\boldsymbol{F}(x, y, z)$ defined, the various level sets of the function are surfaces in the 3D space. In particular, the set of all points at which $\boldsymbol{F}$ constitutes a surface in the 3D space is given as

$$\boldsymbol{F}(x, y, z) = 0. \tag{1}$$

As an example, if $\boldsymbol{F}(x, y, z) \equiv x^2 + y^2 + z^2 - r^2$, then (1) is a sphere of radius r centered at the origin. Representing objects and their surfaces in the implicit form using a mathematical function has a number of advantages. The representation being compact, the surface can be enumerated to any degree of precision, unlike the mesh representations, for example. Tangents and normals to the surface have analytical representations from the gradient of the function $\boldsymbol{F}$. As a result, the synthesis of the image of an implicit object is very straightforward. The general procedure is to send 'eye rays' from the camera to the scene. Then test for intersection is carried out between each ray and the object. If there is an intersection, information around the point of intersection like the surface normal is used to calculate the brightness on the image. The test for ray-object intersections is greatly simplified in the implicit surface case because of the closed form representation.

Some of the commonly used implicit surfaces for object modeling include the quadrics, blobby molecules, meta balls [6], etc. Given the image of an implicit surface, we attempt to extract the 'characteristic parameters' corresponding to the surface. For example, when the image of a sphere is given, we are interested in determining the radius of the sphere, which

is the characteristic parameter. This is in contrast to the traditional approaches where the goal is to directly evaluate the surface gradients. The detailed discussion of the analysis process is given in [7] and described here briefly.

Objects from the same class of implicit surfaces differ from each other in their characteristic parameters. For example, two sphere objects are different if they have different radii. We modify (1) to explicitly include this dependence.

$$\boldsymbol{F}(x, y, z; \boldsymbol{c}) = 0 \tag{2}$$

The vector $\boldsymbol{c}$ represents the characteristic parameters. Under identical imaging conditions, the images $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$ respectively of objects $\boldsymbol{O}_1$ and $\boldsymbol{O}_2$ belonging to the same class will be identical if and only if their characteristic parameter vectors are the same ($\boldsymbol{c}_1 = \boldsymbol{c}_2$). Conversely, if $\boldsymbol{O}_1$ and $\boldsymbol{O}_2$ are different ($\boldsymbol{c}_1 \neq \boldsymbol{c}_2$), there will be an 'error' between $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$. The error will depend on the deviation of $\boldsymbol{c}_2$ from $\boldsymbol{c}_1$. The error can be defined as the sum of the squares of the differences between the pixel values in images $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$.

$$E = \sum_{i,\, j} [\boldsymbol{I}_1(i, j) - \boldsymbol{I}_2(i, j)]^2. \tag{3}$$

Given an image $\boldsymbol{I}$, the goal is to extract the characteristic parameter vector $\boldsymbol{c}$ of the object represented by its image. It is equivalent to determining the $\boldsymbol{c}^*$ for which the error between the original image and the image synthesized from $\boldsymbol{c}$ becomes the minimum. Hence, the task of determining the shape of the object becomes one of searching for the specific $\boldsymbol{c}^*$ which minimizes the error between the original and synthesized images.

The general procedure for finding the vector $\boldsymbol{c}^*$ for the object, given its image $\boldsymbol{I}$, is formulated as—Start with an initial $\boldsymbol{c}$ and synthesize the corresponding image $\boldsymbol{I}'$, progressively refine the parameter vector $\boldsymbol{c}$ such that the error E between $\boldsymbol{I}$ and $\boldsymbol{I}'$ approaches the minimum value 0. The characteristic vector $\boldsymbol{c}^*$ when the error becomes 0 will be the one corresponding to the object in the original image.

The error as function for a simple object like sphere of radius r is shown in Fig. 1. The error is a function of $\boldsymbol{c}^* = r^*$ when $\boldsymbol{c} = r$ is the desired characteristic parameter. The error being quadratic function of r has a minimum as seen in Fig. 1. In general, $\boldsymbol{c}$ will be an n-D vector and therefore has a global minimum.

The synthesis process can be explicitly modeled for the simple implicit surfaces like spheres and ellipsoid, etc. The error has a closed form expression and hence gradient-based optimization methods can be used to minimize the error.

Our ability to carry out this analysis successfully is dependent on the closeness to reality of the original synthesis process. Realistic image synthesis processes incorporate complex camera models, accurate reflectance models involving a large number of parameters, and complicated light source distributions which invariably precludes analytical equations for the synthesis process. Consequently, there is only a 'process' and no 'equation' to model this process. As a result, the traditional analytical gradient-based optimization methods cannot be used. Therefore, we need to resort to optimization methods that involve only the function values and not the derivatives. It must be noted that since we are not constrained

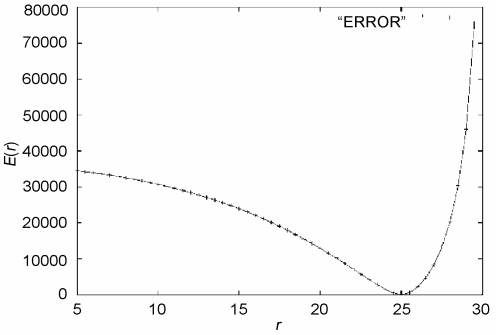

FIG. 1. Squared error between sphere of radius $r = 25$ and synthesized sphere for different radii.

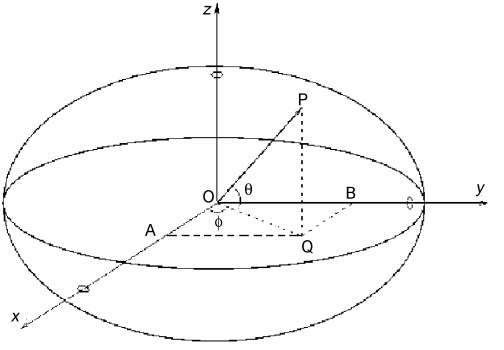

FIG. 2. Elliptic geometry.

by the requirement of explicit model for the realistic synthesis processes, we can allow the synthesis process to be as complex as needed to achieve added realism.

In such situations, the case of a single-point light source with Lambertian surface reflection has been studied extensively [1, 2] for recovering the shape from shading. An important class of implicit surfaces commonly occurring in computer graphics and vision are the quadrics [8]. For simple objects like the quadrics, image synthesis under Lambertian reflectance and single-point light source is analytically deducible. Quadric analysis under these simplistic assumptions is demonstrated in the following section. We shall resort to numerical optimization techniques when an analytical model cannot be derived, or, when derived, is too cumbersome to work with. In this work, we use the polytope method [9] for minimizing the error in the simulations. We illustrate the framework for shape recovery from shading for the quadric surfaces in the following section.

3. Quadric surfaces

If the implicit function $\boldsymbol{F}$ characterizing a surface is a polynomial, then the surface is called an algebraic surface. A quadratic surface or quadric is defined by a second-degree algebraic surface, and is given by the general equation as follows.

$$ax^2 + by^2 + cz^2 + 2(fyz + gzx + hxy) + 2(px + qy + rz) + d = 0. \tag{4}$$

Sphere, ellipsoid, paraboloid and hyperboloid are examples of quadrics. Quadrics and its generalization called 'super-quadrics' are popular models for object representation in computer vision [8].

The equation of an ellipsoid centered at (x_0, y_0, z_0) is given by

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} + \frac{(z - z_0)^2}{c^2} = 1, \tag{5}$$

where a, b and c are the half-lengths of the ellipsoid. When the image of an ellipsoid with a known origin is given, we would like to determine its characteristic parameter vector $\boldsymbol{c} = [a, b, c]$.

Given the θ and ϕ at a point on the ellipsoid centered at origin, the (x, y, z) coordinates and vice versa can be obtained from the following equations

$$x = a\cos\theta\cos\phi, \quad y = b\cos\theta\sin\phi, \quad z = c\sin\theta$$

$$\theta = \sin^{-1}\left(\frac{z}{c}\right), \quad \phi = \tan^{-1}\left(\frac{a}{b}\cdot\frac{y}{x}\right) \tag{6}$$

The next subsection presents the analysis of the image of an ellipsoid object to recover its characteristic parameters. The other quadrics and super-quadrics can be analyzed similar to the ellipsoid.

3.1. *Ellipsoid synthesis*

The synthesis of the object is done using ray tracing [10] by sending out rays from the origin to various points on the camera screen. The intersection of a ray with ellipsoid is calculated followed by surface normal at the point of intersection. The normal is used to calculate the radiance from the surface. Figure 2 shows the ellipsoid geometry with a ray originating from the origin intersecting the ellipsoid at point P and projected on the camera screen.

Let the point of intersection of the ray with ellipsoid be given by

$$\boldsymbol{r} = \boldsymbol{o} + t\boldsymbol{d}, \tag{7}$$

where $\boldsymbol{o}(x_o, y_o, z_o)$ and $\boldsymbol{d}(x_d, y_d, z_d)$ are the origin and the direction vectors of the ray, respectively. The points on the ray can be obtained for $t \geq 0$ extending to infinity by substituting the coordinates of the point in the ellipsoid equation.

$$\frac{(x_0 - t.x_d)^2}{a^2} + \frac{(y_0 - t.y_d)^2}{b^2} + \frac{(z_0 - t.z_d)^2}{c^2} = 1. \tag{8}$$

Rearranging the terms in powers of t, we have

$$At^2 + Bt + C = \mathbf{0}$$

where

$$A = \frac{x_d^2}{a^2} + \frac{y_d^2}{b^2} + \frac{z_d^2}{c^2}, \quad B = \frac{2x_0x_d}{a^2} + \frac{2y_0y_d}{b^2} + \frac{2z_0z_d}{c^2}, \text{ and } C = \frac{x_0^2}{a^2} + \frac{y_0^2}{b^2} + \frac{z_0^2}{c^2} - 1. \tag{9}$$

Solving for t, the ray intersects the ellipsoid for real values and the value closest to origin is chosen. Let t_i be the value of t at point $P_i(x_p, y_p, z_p)$ of intersection. We can get the values of θ and ϕ at the point using (6). The tangents to the surface at point $\boldsymbol{P}$ can be computed by taking the partial derivative with θ and ϕ and the normal by taking the cross product of the tangents as follows:

$$\begin{bmatrix} \dfrac{\partial \boldsymbol{P}}{\partial\theta} \\ \dfrac{\partial \boldsymbol{P}}{\partial\phi} \end{bmatrix} = \begin{bmatrix} -a\sin\theta\cos\phi & -b\sin\theta\sin\phi & c\cos\theta \\ -a\cos\theta\sin\phi & b\cos\theta\cos\phi & 0 \end{bmatrix},$$

$$\boldsymbol{N} = \frac{\dfrac{\partial \boldsymbol{P}}{\partial \theta} \times \dfrac{\partial \boldsymbol{P}}{\partial \phi}}{\left| \dfrac{\partial \boldsymbol{P}}{\partial \theta} \times \dfrac{\partial \boldsymbol{P}}{\partial \phi} \right|} \tag{10}$$

A local coordinate system can be defined about the point using (10).

3.2. *Ellipsoid analysis under Lambertian reflectance model*

The ellipsoid analysis for the analytically tractable case of Lambertian reflection under a single-point light source is presented in this section [11]. Minimization of the error in this case is done using gradient-based optimization methods. Under more realistic imaging conditions, numerical methods are employed for performing the multiparameter optimization. First, we derive the necessary equations for performing optimization under the Lambertian reflectance and a single-point light source.

Given the x and y values at any point on the surface of ellipsoid, the corresponding value of z can be calculated from the ellipsoid equation (5) as

$$z = z_0 \pm c\sqrt{1 - \frac{(x - x_0)^2}{a^2} - \frac{(y - y_0)^2}{b^2}}. \tag{11}$$

Hence a point $\boldsymbol{P}$ on the surface of the ellipsoid is given by (x, y, z) where z is given by (11) with positive sign for convenience. The tangents to the surface at point $\boldsymbol{P}$ can be computed by taking the partial derivative with x and y and the normal by taking the cross product of the tangents as follows:

$$\begin{bmatrix} \dfrac{\partial \boldsymbol{P}}{\partial x} \\ \dfrac{\partial \boldsymbol{P}}{\partial y} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dfrac{-\dfrac{c}{a}\left(\dfrac{x - x_0}{a}\right)}{\sqrt{1 - \dfrac{(x - x_0)^2}{a^2} - \dfrac{(y - y_0)^2}{b^2}}} \\ 0 & 1 & \dfrac{-\dfrac{c}{b}\left(\dfrac{y - y_0}{b}\right)}{\sqrt{1 - \dfrac{(x - x_0)^2}{a^2} - \dfrac{(y - y_0)^2}{b^2}}} \end{bmatrix},$$

$$\boldsymbol{N} = \frac{\partial \boldsymbol{P}}{\partial x} \times \frac{\partial \boldsymbol{P}}{\partial y} = \left[\frac{1}{a}\left(\frac{x - x_0}{a}\right) \ \frac{1}{b}\left(\frac{y - y_0}{b}\right) \ \frac{1}{c}\left(\frac{z - z_0}{c}\right) \right] \tag{12}$$

If the single-point light source is situated at (x_l, y_l, z_l) in the world, then the light-source direction with respect to point $\boldsymbol{P}$ is

$$\boldsymbol{L} = (x_l - x, y_l - y, z_l - z).$$

In the simplified case of Lambertian reflectance, the brightness from point $\boldsymbol{P}$ is independent of the viewing direction and is dependent only on the $\boldsymbol{N}$ and $\boldsymbol{L}$ directions. Assuming Lam-

bertian reflection with a single-point light source at infinity, the brightness due to a point on the object surface is

$$R = \alpha \cos\theta, \tag{13}$$

where θ is the angle between the surface normal vector $\boldsymbol{N}$ and the light source vector $\boldsymbol{L}$, and α is the scaling factor. In the case of the ellipsoid, R is dependent on the characteristic parameters by virtue of the fact that $\boldsymbol{N}$ on the surface of the ellipsoid at any point (x, y, z) is dependent on a, b and c values. The expression for R can be written as

$$R = \frac{\boldsymbol{f}(a,b,c)}{\boldsymbol{h}(a,b,c)\ \boldsymbol{g}(x,y,z)}$$

where

$$\boldsymbol{f}(a,b,c) = \frac{1}{a}\left(\frac{x-x_0}{a}\right)(x_l - x) + \frac{1}{b}\left(\frac{y-y_0}{b}\right)(y_l - y) + \frac{1}{c}\left(\frac{z-z_0}{c}\right)(z_l - z),$$

$$\boldsymbol{h}(a,b,c) = \sqrt{\frac{1}{a^2}\left(\frac{x-x_0}{a}\right)^2 + \frac{1}{b^2}\left(\frac{y-y_0}{b}\right)^2 + \frac{1}{c^2}\left(\frac{z-z_0}{c}\right)^2},$$

$$\boldsymbol{g}(x,y,z) = \sqrt{(x_l - x)^2 + (y_l - y)^2 + (z_l - z)^2} \tag{14}$$

The expression for R can be used to compute the characteristic parameter vector $\boldsymbol{c} = [a, b, c]$ of an ellipsoid represented by its image $\boldsymbol{I}$. The error between the image $\boldsymbol{I}$ and a synthesized image $R(a, b, c)$ will be a function of a, b, and c and is given in the mean square sense by

$$E(a,b,c) = \sum_{i,\ j} [\boldsymbol{R}_{i,j}(a,b,c) - \boldsymbol{I}_{i,j}]^2. \tag{15}$$

Because of the simple form of Lambertian reflection, we get a closed form expression for E. E is minimized by employing any of the widely used techniques for multivariable optimization which involve the partial derivatives $\partial E/\partial a$, $\partial E/\partial b$, and $\partial E/\partial c$. Gauss–Newton method is an optimization method well suited for minimizing functions of the form

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_i e_i^2(\boldsymbol{w}) = \frac{1}{2} e^T(\boldsymbol{w}) e(\boldsymbol{w}), \tag{16}$$

where $\boldsymbol{w} = [w_1\ w_2 \cdots w_n]^T$ is n-dimentional vector.

The Gauss–Newton iteration procedure [12] in the case of ellipsoid is presented as follows. The error term $e_i(\boldsymbol{w})$ in (16) can be expanded in the Taylor series around a point $\boldsymbol{w}_n$, and $\boldsymbol{e}(\boldsymbol{w})$ can written in matrix form

$$\boldsymbol{e}(\boldsymbol{w}) = \boldsymbol{e}(\boldsymbol{w}_n) + \boldsymbol{J}(\boldsymbol{w}_n)(\boldsymbol{w} - \boldsymbol{w}_n),$$

where $$\boldsymbol{e}(\boldsymbol{w}_n) = [e_1(\boldsymbol{w}_n)\ e_2(\boldsymbol{w}_n) \cdots e_m(\boldsymbol{w}_n)]^T,$$

$$\boldsymbol{J}(\boldsymbol{w}_n) = \begin{bmatrix} \frac{\partial e_1}{\partial a} & \frac{\partial e_1}{\partial b} & \frac{\partial e_1}{\partial c} \\ \frac{\partial e_2}{\partial a} & \frac{\partial e_2}{\partial b} & \frac{\partial e_2}{\partial c} \\ \vdots & \vdots & \vdots \\ \frac{\partial e_m}{\partial a} & \frac{\partial e_m}{\partial b} & \frac{\partial e_m}{\partial c} \end{bmatrix}_{\boldsymbol{w}=\boldsymbol{w}_n}, \tag{17}$$

$\boldsymbol{J}(\boldsymbol{w}_n)$ is the Jacobian of the error vector $\boldsymbol{e}(\boldsymbol{w})$ at $\boldsymbol{w} = \boldsymbol{w}_n$. The mean square error in (16) is iteratively minimized. The update equation for the $\boldsymbol{w}_n$ is obtained by differentiating $E(\boldsymbol{w})$ with respect to $\boldsymbol{w}$ and setting it to zero, and solving the resulting equation.

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - [\boldsymbol{J}^T(\boldsymbol{w}_n)\boldsymbol{J}(\boldsymbol{w}_n)]^{-1}\boldsymbol{J}^T\boldsymbol{e}(\boldsymbol{w}_n). \tag{18}$$

In order to employ the Gauss–Newton method, the matrix product $\boldsymbol{J}^T(\boldsymbol{w})\boldsymbol{J}(\boldsymbol{w})$ must be nonsingular at each intermediate point $\boldsymbol{w}_n$ during the iteration. It is a symmetric matrix and hence positive semidefinite. To ensure that it is nonsingular, it is customary to add a diagonal matrix $\delta\boldsymbol{I}$ to $\boldsymbol{J}^T(\boldsymbol{w})\boldsymbol{J}(\boldsymbol{w})$, where δ is a small positive constant. The resulting matrix is positive definite and hence always invertible. In practice, we start with a δ value and progressively decrease it as the number of iterations increases.

To apply the Gauss–Newton method for ellipsoid analysis, we have to obtain the derivative of R with respect to a, b, and c, which in turn requires evaluation of partial derivatives of $\boldsymbol{f}$, $\boldsymbol{h}$, and $\boldsymbol{g}$ with respect to ellipsoid parameters (a, b, and c). These partial derivatives involve partial derivatives of z with respect to a, b, and c with x and y in the ellipsoid equation being fixed. These partial derivatives can be obtained using (14).

For more complex imaging scenarios (for example, 'copper' ellipsoid under two-point light sources), the expression for R will not be as simple as in (13). Ideally, we would like to work with natural images, where this simplified assumption will not suffice. In the general case of realistic reflectance models and more general lighting conditions, the absence of an explicit image irradiance equation constrains us to work only with pixel values in synthesized images. Therefore, in the absence of an error function, we have to resort to numerical optimization techniques for minimizing the error between two images to find the characteristic parameter. In the case of the ellipsoid, the error is a function of the three variables a, b and c. Consequently, numerical techniques for multivariable function optimization must be employed.

The Nelder–Mead Downhill Simplex Method [8], otherwise called the polytope method, is a well-known numerical algorithm for optimizing functions defined over multiple variables. The method proceeds by moving a simplex, which is a polygon of $n + 1$ sides when minimization is done over n variables, in a specific manner in order to bracket the minimum error point. The search is continued by progressively reducing the size of the simplex till the minimum error point is attained within an acceptable tolerance. This simple algorithm is time consuming but is robust. An elaborate discussion of this algorithm can be found in [9].

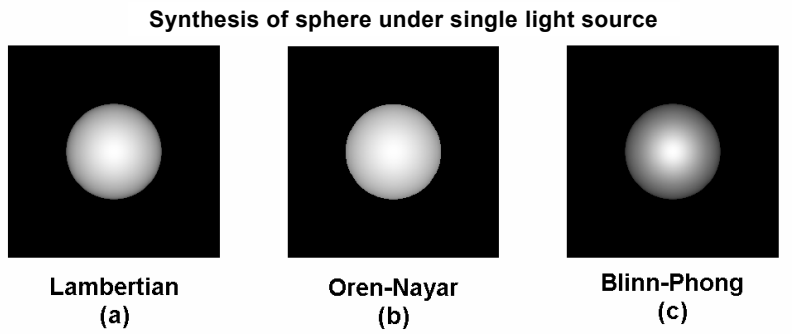

FIG. 3. Sphere generated under one light source with different reflectance models.

3.3. *Simulations*

In the simulations, we consider the synthesis of sphere followed by the ellipsoid under different imaging conditions (light sources and reflectance models) in illustrating the analysis by synthesis of quadric surfaces. Reflectance models range from simple one-parameter models which are coarse approximations of reflection to more complex and accurate physics-based models incorporating almost every phenomenon affecting surface reflection. We consider the Lambertian, Oren–Nayar [13], Blinn-Phong [14] and Cook-Torrance [15] reflectance models for the simulations. The Lambertian and Oren–Nayar models explain the process of diffuse reflection exhibited by dull, matte surfaces. Surfaces exhibiting Lambertian reflection appear equally bright from all viewing directions as they reflect light with equal intensity in all directions. Lambertian reflection is therefore independent of the viewing direction and is dependent only on the angle between the light-source direction and the surface normal. The Oren–Nayar model is a generalization of the Lambertian reflectance model and is able to account for complex geometric and radiometric phenomena such as masking, shadowing and interreflections. The Blinn–Phong model is an empirical model for specular reflection in which the specular highlight is modeled by an exponentiated cosine. The model due to Cook and Torrance is an example of a physics-based specular reflection model where the reflectance function is derived by analyzing the physical processes underlying reflection. An extensive discussion of the various reflectance models can be found in [10].

Figure 3 show the synthesis of the sphere generated under single light source with Lambertian, Oren–Nayar [13] and Blinn–Phong [14] reflectance models. Figure 4 shows the sphere generated with two light sources for Blinn–Phong and Cook–Torrance [15] reflectance models and Cook–Torrance reflectance model under three light sources. It may be noted that the treatment in sub-sections 3.1 and 3.2 can be easily reduced to the sphere where the characteristic vector is $\boldsymbol{c} = r$, the radius of the sphere.

Figures 5 and 6 show the synthesis of an ellipsoid under the identical imaging conditions as in the case of sphere in Figs 3 and 4, respectively. It can be observed in both the cases that the fall in the brightness towards the edges is less pronounced in the case of Lamber-tain model compared to the Oren–Nayar model which is one of the characteristics of the Oren–Nayar model for rough surfaces. In Figs 4 and 6, the objects have been synthesized under the presence of specular components. The specular model in Figs 4(a) and 6(a) is the em-

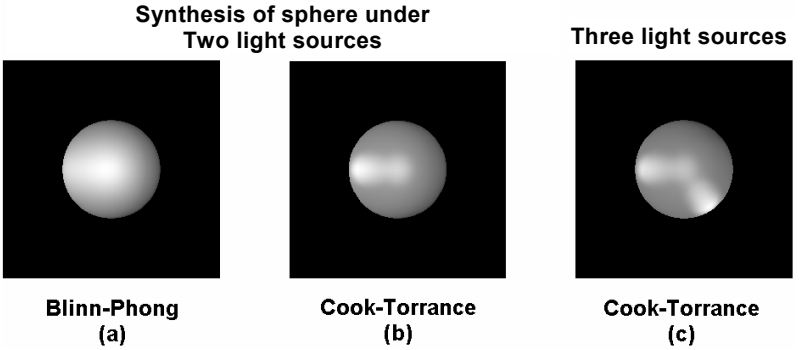

FIG. 4. Sphere generated under two and three light sources with different reflectance models.

pirical Blinn–Phong model while the more accurate Cook–Torrance model has been employed for the specular component in Figs 4(b) and 6(b) as well as Figs 4(c) and 6(c). Thus, it is seen that under all these different imaging scenarios, the proposed method can be applied to extract the characteristic vector of any implicit surface accurately.

4. Analysis by synthesis of general surfaces

Modeling of surfaces is a major area of study in the field of geometric modeling. Computer graphics engineers have devised number of tools to model surfaces for computer-aided geometric design and efficient algorithms to render these general surfaces. The objective in this section is to demonstrate the utility of the proposed analysis by synthesis framework for determining the shape information from shading in images of general surfaces. As a first step, we need an efficient and 'good' representation scheme or model for the surfaces to synthesize the surface for visualization. Bézier surfaces are one of the most popular representations for describing surfaces. These are formed by the linear combinations of Bernstein polynomials [16].

4.1. *Bézier surfaces*

Bézier techniques provide a geometric-based method for describing and manipulating polynomial curves and surfaces represented in parametric form. The power of Bézier techniques

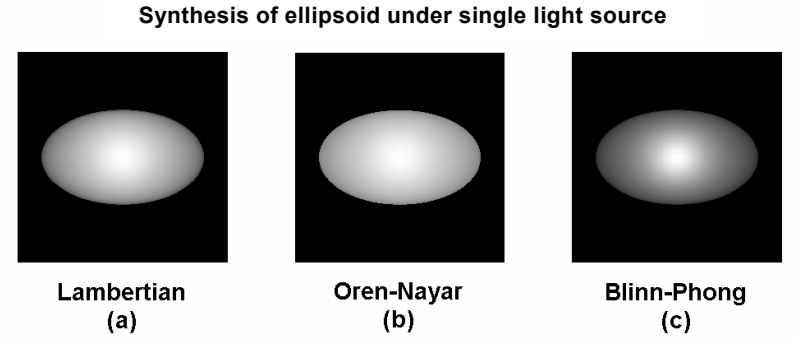

FIG. 5. Ellipsoid generated under one light source with different reflectance models.

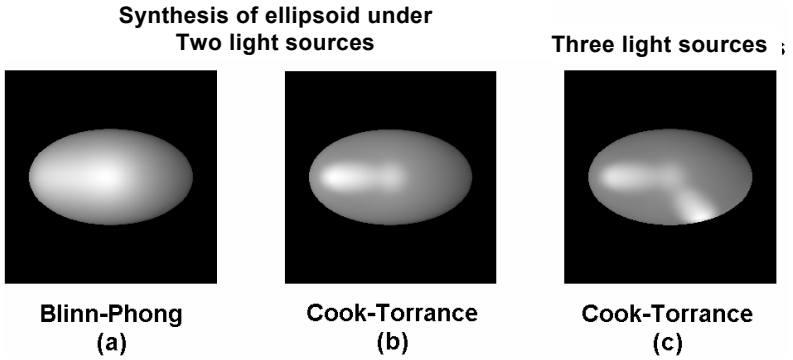

FIG. 6. Ellipsoid generated under two and three light sources with different reflectance models.

lies in their ability to provide a highly intuitive basis for understanding the curves and surfaces. Because of the twin advantages of the mathematical sophistication and intuitive form, Bézier methods are popular in computer graphics and computer-aided geometric design for dealing with curves and surfaces [16]. The Bézier curves are introduced first and extended to the Bézier surfaces.

Polynomial parametric curves are widely used for representing curves. A curve of degree n is represented with polynomial coordinates as

$$\boldsymbol{P}(t) = \boldsymbol{p_n} t^n + \boldsymbol{p}_{\mathrm{n}-1} t^{n-1} + \cdots + \boldsymbol{p}_1 t + \boldsymbol{p}_0, \tag{19}$$

where $\boldsymbol{P}(t)$ is the vector $[x(t), y(t), z(t)]^T$ and $\boldsymbol{p}_i$ are control points. The monomial form of polynomial curves is not the only representation for polynomial curves. There are other representations available which are more geometrically intuitive. The Bézier formulation built on Bernstein basis functions provides an intuitive framework for representing curves. A Bézier curve of degree n is of the form

$$\boldsymbol{P}(t) = \sum_{i=0}^{n} \boldsymbol{P}_i B_{i,\,n}(t) = \boldsymbol{P^T B},$$

where $B_{i,\,n}(t)$ are degree n Bernstein polynomials

$$B_{i,\,n}(t) = \binom{n}{i} t^i (1-t)^{n-i} \quad \text{and} \quad \binom{n}{i} = \frac{n!}{i!(n-i)!}. \tag{20}$$

The geometric intuition of the Bézier curve representation is due to the fact that the coefficients $\boldsymbol{P}_i$ of the Bernstein polynomials in (20) are themselves points in 3D space. Bernstein polynomials possess some important properties [16] which make them extremely useful in geometric modeling and computer graphics.

Among the Bézier curves, the cubic Bézier curves are more often used than any other degree. These curves have a reasonably low degree of 3 and at the same time are flexible to a certain extent. Given the control points $\boldsymbol{P}_0$, $\boldsymbol{P}_1$, $\boldsymbol{P}_2$, $\boldsymbol{P}_3$, the cubic Bézier curve and the corresponding Bernstein polynomials are defined by (20) for $n = 3$. The cubic Bézier curve can be written in matrix form by grouping together the terms in powers of t which is convenient for performing certain operations on the curve.

$$\boldsymbol{P}(t) = [1 \quad t \quad t^2 \quad t^3]\boldsymbol{B}'[\boldsymbol{P}_0 \quad \boldsymbol{P}_1 \quad \boldsymbol{P}_2 \quad \boldsymbol{P}_3]^T \tag{21}$$

where the elements $b_{i,j}$ of matrix $\boldsymbol{B}'$ are coefficients of power basis used to determine the respective Bernstein polynomials.

Matrix $\boldsymbol{B}'$ is the blending function for $\boldsymbol{P}(t)$ facilitating the operations on the curve. For example, it is often required to subdivide the curve into two parts, one where t ranges from 0 to 1/2 and the other for t ranging from 1/2 to 1. This operation can be done under the matrix representation as follows:

$$\boldsymbol{P}(t) = [1 \quad t \quad t^2 \quad t^3]\boldsymbol{B}'(\boldsymbol{S}_{\left[0,\frac{1}{2}\right]} + \boldsymbol{S}_{\left[\frac{1}{2},1\right]})[\boldsymbol{P}_0 \quad \boldsymbol{P}_1 \quad \boldsymbol{P}_2 \quad \boldsymbol{P}_3]^T. \tag{22}$$

Thus, by grouping the $\boldsymbol{S}$ matrix and the vector of points together and multiplying them, the subdivision process can be seen as one of defining a new curve by modifying the existing control points. The subdivision operation is very useful when finding the intersection of a line with the Bézier curve. To find the point of intersection, the curve is divided into two parts following the above discussion, and the half of the curve whose bounding box is intersected by the line is again subdivided. This bounding box intersection and subdivision process can be carried to a desired level of resolution after which the mid-point of the smallest bounding box intersected by the line is taken to be the point of intersection. The subdivision procedure is used to render Bézier curves as well as Bézier patches.

4.2. *Bézier patches*

The extension of Bézier curves to surfaces is called the Bézier patch [16]. A parametric surface is the result of a map of the real plane (domain) into the 3D space (co-domain). If a (u, v)-coordinate system is defined on this real plane, then $\boldsymbol{P}(u, v)$ is a point on a surface in 3D and its x, y, and z coordinates are functions of u and v. A Bézier patch is constructed from an $m \times n$ array of control points $\{\boldsymbol{P}_{i;j} : 0 \le i \le m, 0 \le j \le n\}$. The collection of control points is called the control net. Given the control net, the Bézier patch is defined as

$$\boldsymbol{P}(u,v) = \sum_{j=0}^{n}\sum_{i=0}^{m} \boldsymbol{P}_{i,j} B_{i,m}(u) B_{j,n}(v)$$

$$= [B_{0,m}(u)\ B_{1,m}(u) \cdots B_{m,m}(u)]\boldsymbol{P}^T[B_{0,n}(v)\ B_{1,n}(v) \cdots B_{n,n}(v)]^T. \tag{23}$$

The Bézier patch has the same general form as the Bézier curve, with the summation done over a control matrix instead of a control vector. The bivariate Bernstein polynomials serve as the blending functions for the control points. They are simply the product of any two univariate Bernstein polynomials. The Bézier patch is shown in Fig. 7.

The cubic Bézier surfaces can be written in matrix form by grouping together the terms in powers of u and v for subdivision of patches as done in the case of Bézier curves. The resulting matrix equation is

$$\boldsymbol{P}(u,v) = [1 \quad u \quad u^2 \quad u^3]\boldsymbol{B}''\boldsymbol{P}\boldsymbol{B}''^T[1 \quad v \quad v^2 \quad v^3] \tag{24}$$

where $\boldsymbol{B}''$ are coefficients of powers of u and v.

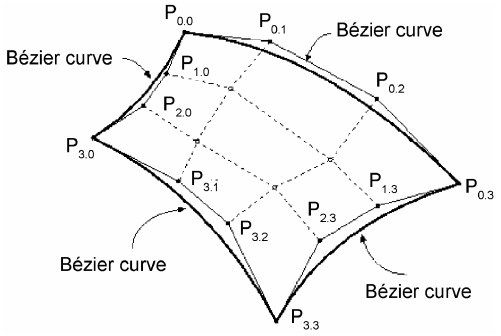

FIG. 7. Bézier patch.

Subdivision of a Bézier patch $\boldsymbol{P}(u, v)$ into subpatches is done much the same way as Bézier curves. Both the u and v parameters can be split into two parts each, [0, 1/2] and [1/2, 1], resulting in four Bézier subpatches. The control net of the four subpatches is obtained by operating $\boldsymbol{S}$, in a similar way as in (22) on the original Bézier patch's control net $\boldsymbol{P}$. The control nets of the subpatches are

$$\boldsymbol{P}_1 = \boldsymbol{S}_{\left[0,\frac{1}{2}\right]}\boldsymbol{P}\boldsymbol{S}^T_{\left[0,\frac{1}{2}\right]}, \quad \boldsymbol{P}_2 = \boldsymbol{S}_{\left[0,\frac{1}{2}\right]}\boldsymbol{P}\boldsymbol{S}^T_{\left[\frac{1}{2},1\right]},$$

$$\boldsymbol{P}_3 = \boldsymbol{S}_{\left[\frac{1}{2},1\right]}\boldsymbol{P}\boldsymbol{S}^T_{\left[0,\frac{1}{2}\right]}, \quad \boldsymbol{P}_4 = \boldsymbol{S}_{\left[\frac{1}{2},1\right]}\boldsymbol{P}\boldsymbol{S}^T_{\left[\frac{1}{2},1\right]}, \tag{25}$$

where $\boldsymbol{P}_i$ correspond to the control net of $\boldsymbol{B}_i$.

4.3. *Synthesis of Bézier surfaces*

The major task in image synthesis by ray tracing is to find the point of intersection of a ray from the camera and the object in the scene. The normal at the point of intersection will then be used to calculate the radiance from the surface point in the camera direction. Unlike the direct process of calculation of the intersection point due to closed form of the object equations in the case of implicit surfaces, there is no parallel procedure to calculate the intersection point(s) in the case of parametric surfaces like the Bézier patches. Elaborate work has been carried out in the field of computer graphics to do efficient synthesis of parametric surfaces. Ray tracing techniques for parametric surfaces fall into two major categories, namely, subdivision of numerical methods. Subdivision methods rely on identifying regions in methods and the scene for performing ray–surface intersections in an efficient manner. The vocabulary in the subdivision-based methods therefore is similar to that of the physical entities in the system. The numerical methods on the other hand treat the intersection problem as one of optimizing a function of the ray and surface parameters. Early attempts at ray tracing on parametric surfaces were all subdivision based. In these methods, the ray is intersected with the bounding volume of the surface by iteratively subdividing the surface patch into smaller subpatch till the ray intersection with the bounding volume of the subpatch is found. Some of the bounding volumes used in earlier works include spheres, axis-aligned bounding boxes and parallelepipeds. Toth [17] has proposed a numerical method for ray tracing Bézier surfaces. In this method, the multivariate Newton iteration has been employed to find ray–surface intersections. In this paper, we present a hybrid algorithm for the Bézier surface synthesis drawn from Toth [17].

Let $\boldsymbol{H}(u, v) = [x(u, v); y(u, v); z(u, v)]^T$ be a parametric surface with parameters u and v taking values in [0; 1]. It is required to find the intersection of $\boldsymbol{H}(u, v)$ with a ray $\boldsymbol{R}(t)= \boldsymbol{o} + t\boldsymbol{d}$. The ray, parameterized by t, has origin $\boldsymbol{o}$ and direction $\boldsymbol{d}$. An intersection between $\boldsymbol{H}(u, v)$ and $\boldsymbol{R}(t)$ occurs if and only if $f(x) = \boldsymbol{H}(u, v) + \boldsymbol{R}(t) = 0$, and $x = [u, v, t]^T$. To solve these nonlinear equations in multiple variables, the Newton scheme is commonly employed. If $\boldsymbol{Y}$ is any 3×3 matrix, then the Newton iteration scheme is given by $x_{k+1} = x_k + \boldsymbol{Y}f(x_k)$. The vector $\boldsymbol{Y}f(x_k)$ is referred to as the Newton step. When $\boldsymbol{Y}$ is the inverse Jacobian of f at x_k, then geometrically it represents the coordinates of the intersection of the ray and the tangent plane at x_k. Normally, matrix $\boldsymbol{Y}$ is updated in each iteration resulting in quadratic convergence. When it is held constant, the resulting iteration scheme is called simple Newton iteration, and has linear convergence.

If the initial guess of the solution is 'good', then it suffices to perform the simple Newton scheme. The major focus of Toth's work [17] is to arrive at such an initial value to start the iteration scheme. The apparatus of interval mathematics has been used in his work. Toth states and proves two theorems which give conditions for safely starting the Newton iteration. This is done by defining an operator called the Krawczyk's operator which is a box in parameter space. Let $\boldsymbol{X}$ be a box contained in an open set of interest in 3D Euclidean space. For any real vector $\boldsymbol{y}$ in $\boldsymbol{X}$, and a nonsingular 3×3 matrix $\boldsymbol{Y}$, $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{Y})$, called the Krawczyk's operator, is defined. Krawczyk's operator constrains the movement of elements of $\boldsymbol{X}$ under a Newton step performed with $\boldsymbol{Y}: \forall x \in \boldsymbol{X}$, the value $x - \boldsymbol{Y}f(x) \in \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{Y})$. In addition, all solutions to $f(x) = 0$ in $\boldsymbol{X}$ are in $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{Y})$. Using the Krawczyk's operator, two criteria for identifying 'safe' regions for starting the simple Newton iteration are defined by Toth. When a region $\boldsymbol{X}$ in parameter space satisfies one of these criteria, the simple Newton iteration is guaranteed to converge to a single solution for $f(x) = 0$ from any starting point in $\boldsymbol{X}$.

The subdivision schemes as well as the numerical approaches have their own advantages and disadvantages. The main advantage of the subdivision methods is their simplicity. The intersection calculations for the bounding volumes are relatively less computation intensive. Also, the subdivision of a patch into subpatches is a standard procedure and, therefore, the hierarchy of subdivisions can be pre-computed before the actual synthesis process starts. But the disadvantage of these methods is that the intersection calculations typically have to be done for a large number of bounding volumes before the procedure ends. The numerical methods on the other hand typically converge in a few iterations, though convergence issues are to be taken care of a priori. The disadvantages of the numerical methods include starting the procedure with a good initial estimate, numerical stability issues, etc. The obvious way to proceed is to combine these methods to improve the synthesis of parametric surfaces. Toth's method discussed earlier has subdivision as part of its ray–surface intersection calculation. Lischinski and Gonczarowski [18] made a few modifications to Toth's method to improve synthesis efficiency. We have made use of this method in our work for Bézier surface synthesis.

4.4. *Analysis of Bézier surfaces*

Bézier surfaces offer a better approximation to naturally occurring surfaces than implicit surfaces. A Bézier surface is completely determined by its control net. The control net in

the case of cubic Bézier surfaces has 16 control points in three dimensions. Therefore, the characteristic vector for a Bézier surface has 48 components.

Given the image of a Bézier surface, our task is to extract its characteristic vector. For an arbitrary characteristic vector $\boldsymbol{c}$, the error $\boldsymbol{E}(\boldsymbol{c})$ is calculated by first synthesizing the image of the Bézier surface with characteristic vector $\boldsymbol{c}$ followed by a sum of squared differences between the pixels of the original and synthesized images. The Nelder–Mead Downhill Simplex method is employed for determining the true characteristic vector of the Bézier surface in the original image. The Downhill Simplex method as applied to Bézier surface analysis is as follows.

Algorithm: Nelder–Mead Simplex method for Bézier surface analysis

INITIALIZATION–Construct a simplex in $\mathfrak{R}^{48}$ represented by its vertices $\boldsymbol{c}^0$, $\boldsymbol{c}^1$, $\boldsymbol{c}^2$, …, $\boldsymbol{c}^{48}$. Choose a stop criterion $\varepsilon > 0$.

Step 1. Reorder the vertices such that the errors are ordered as $\boldsymbol{E}(\boldsymbol{c}^i) \geq \boldsymbol{E}(\boldsymbol{c}^{i+1})$, $i = 0, 1, 2, \ldots, 48$.

Step 2. Find the least i such that $\boldsymbol{E}(\boldsymbol{c}^i) > \boldsymbol{E}(\boldsymbol{d}^i)$ where $\boldsymbol{d}^i = 2\,\dot{\boldsymbol{c}}^i - \boldsymbol{c}^i$ is the reflection of $\boldsymbol{c}^i$ and $\dot{c}^i = \frac{1}{48}\sum_{j \neq i} c^i$. If such i exists, replace $\boldsymbol{c}^i$ with $\boldsymbol{d}^i$ and go to Step 1.

Step 3. If the width of the current simplex is less than ε, then stop and report minimum error point of simplex.

Step 4. Set $\boldsymbol{c}^i = 1/2(\boldsymbol{c}^i + \boldsymbol{c}^n)$. This step results in the shrinking of the simplex toward the minimum error vertex. Go to Step 1.

4.5. *Simulations*

We consider the synthesis of Bézier sheet under different imaging conditions (light sources and reflectance models) in illustrating the analysis by synthesis of general surfaces. The imaging conditions are the same as in Section 3. The surface reflectance parameter values used are those of copper.

Figure 8 shows a Bézier sheet rendered under single light source with simple Lambertian model, and more accurate Oren–Nayar and Blinn–Phong reflectance models. Figure 9 shows the Bézier sheet rendered with two light sources for the Blinn–Phong and Cook–Torrance specular reflectance models and the Cook–Torrance reflectance model under three light sources. Thus, the simulation shows that under all these different imaging scenarios,

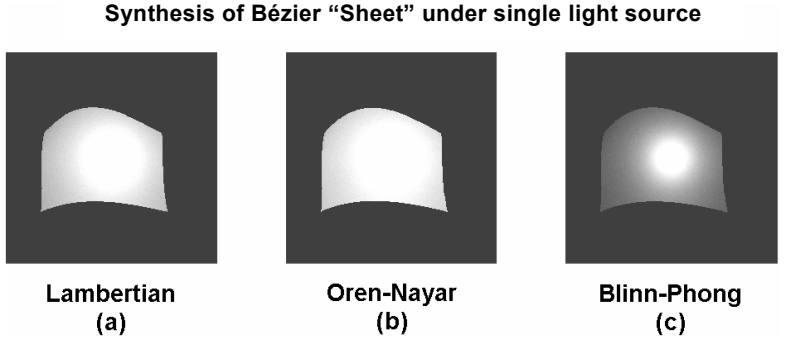

FIG. 8. Bézier sheet generated under one light source with different reflectance models.

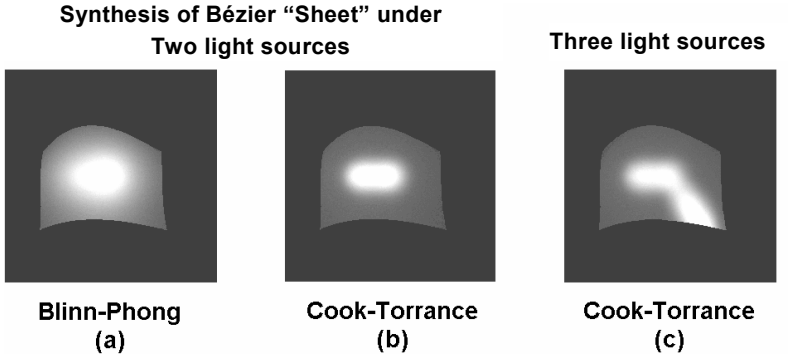

FIG. 9. Bézier sheet generated under two and three light sources with different reflectance models.

the proposed method can be applied to extract accurately the characteristic vector of any general surface.

5. Conclusion

In this paper, we have presented a novel framework for solving the shape-from-shading problem. The main aim was to develop a framework for the shape-from-shading problem under general imaging conditions. Almost all the earlier methods made three constraining assumptions, one each on the camera model (orthographic projection), the light source (single-point source at infinity) and the reflectance model (Lambertian). Consequently, even with complete knowledge of the environment under which an image was created, the earlier methods were less satisfactory in producing surface descriptions. Invariably, these methods relied on modeling the image formation process view to invert it mathematically. The invertibility condition places constraints on the imaging conditions. The proposed analysis-by-synthesis framework moves the complexity from the analysis to synthesis side.

The basic idea behind our approach is to make use of repetitive synthesis of images. The parameters of the synthesis are guided at each step by the error between the original image and the currently synthesized image. The first stage is to decide on a model for representing surfaces. We have demonstrated the analysis-by-synthesis framework for two implicit surface representations: sphere and ellipsoid. The characteristic vector has been introduced to distinguish surfaces within a class. For both these surfaces, an analytic formulation was derived under the Lambertian reflectance and point-light source assumption. Due to the inadequacies of the analytic formulation, a gradient-free numerical formulation was subsequently introduced, first for the sphere, and then for the ellipsoid.

For including more general surfaces under this framework, the Bezier surfaces were used to model the surfaces. The analysis-by-synthesis for the Bezier surfaces was carried out using the Downhill Simplex method. For the Bezier surface, its control net was used to construct the characteristic vector. The utility of the present framework lies in the fact that it lends itself to natural image descriptions. These image descriptions will be incorporated in the synthesis part, which is why the procedure is more tractable than the existing approaches. A potential application scenario for the proposed framework is in indexing of

human faces for the purpose of recognition. To perform the task of recognition of human faces, the first step is to have a corpus of faces and create a 'characteristic vector of the face' for each face. This might be done by first segregating piecewise continuous regions of the face and creating a characteristic vector for each region. Then, given a new face image to be matched with the faces in the corpus, the problem becomes one of matching the characteristic vectors. The same technique can be extended to the problem of matching other object categories too. A recent work on facial modeling makes use of shape-from-shading to recover the facial shape [19]. In this work, a statistical facial model is embedded within an SfS algorithm to recover facial shape with fine local surface details.

There is a growing convergence between the areas of computer graphics and vision. It is hoped that this paper will be a step in that direction.

References

1. B. K. P. Horn, Understanding Image Intensities, *Artif. Intell.*, **8**, 201–231 (1977).
2. Q. Zheng, and R. Chellappa, Estimation of illumination direction, albedo and shape from shading, *IEEE Trans. Pattern Analysis Mach. Intell.*, **13**, 680–702 (1991).
3. M. S. Langer, and S. W. Zucker, Shape-from-shading on a cloudy day, *J. Opt. Soc. Am.*, **11**, 467–478 (1994).
4. K. M. Lee, and C.-C. J. Kuo, Shape from shading with perspective projection, *Computer Vision, Graphics, Image Processing: Image Understanding*, **59**, 202–212 (1994).
5. H. Schultz, Retrieving shape information from multiple images of a specular surface, *IEEE Trans. Pattern Analysis Mach. Intell.*, **16**, 195–201 (1994).
6. A. Opalach, and S. Maddock, An overview of implicit surfaces, in *Introduction to modelling and animation using implicit surfaces*, Course Notes, No. 3, Computer Graphics Int., Leeds, UK, pp. 1.1–1.13 (1995).
7. S. J. Sathish, Shape from shading: analysis by synthesis, M. Sc. (Engng) dissertation, Department of Electrical Engineering, Indian Institute of Science, Bangalore, India (2004).
8. A. Jaklic, A. Leonardis, and F. Solina, *Segmentation and recovery of superquadrics*, *Computational Imaging and Vision*, Vol. 20, Kluwer (2000).
9. S. S. Rao, *Optimization: Theory and applications*, Wiley Eastern (1979).
10. A. S. Glassner, *Principles of digital image synthesis*, Vol. 2, Morgan Kaufman (1995).
11. K. Rajgopal, and S. J. Sathish, Shape from shading: analysis of implicit surfaces by synthesis. *Proc. Int. Conf. Signal Processing* (*ICSP '04*), pp. 1243–1246 (2004).
12. S. Haykin, *Neural networks: A comprehensive foundation*, 2nd edn, Prentice-Hall (1998).
13. S. K. Nayar, K. Ikeuchi, and T. Kanade, Surface reflection: Physical and geometric perspectives, *IEEE Trans. Pattern Analysis Mach. Intell.*, **13**, 611–634 (1991).
14. J. F. Blinn, Models of light reflection for computer synthesized pictures, *SIGGRAPH 77*, pp. 192–198 (1977).
15. R. L. Cook, and K. E. Torrance, A reflectance model for computer graphics, *ACM Trans. Graphics*, **1**, 7–24 (1982).
16. D. Salomon, *Curves and surfaces for computer graphics*, Springer-Verlag (2005).
17. D. L. Toth, On ray tracing parametric surfaces, *Computer Graphics* (*Proc. SIGGRAPH '85*), **19**, 171–179 (1985).
18. D. Lischinski, and J. Gonczarowski, Improved techniques for ray tracing parametric surfaces, *Visual Computer*, **6**, 134–152 (1990).
19. William A. P. Smith, and Edwin R. Hancock, Recovering facial shape using a statistical model of surface normal direction, *IEEE Trans. Pattern Analysis Mach. Intell.*, **28**, 1914–1930 (2006).