Journal of Scientific Computing, Vol. 19, Nos. 1–3, December 2003 (© 2003)

Estimation of 3D Surface Shape and Smooth Radiance from 2D Images: A Level Set Approach

Hailin Jin,[1] Anthony J. Yezzi,[2] Yen-Hsi Tsai,[3] Li-Tien Cheng,[4] and Stefano Soatto[5]

Received May 3, 2002; accepted (in revised form) September 14, 2002

We cast the problem of shape reconstruction of a scene as the global region segmentation of a collection of calibrated images. We assume that the scene is composed of a number of smooth surfaces and a background, both of which support smooth Lambertian radiance functions. We formulate the problem in a variational framework, where the solution (both the shape and radiance of the scene) is a minimizer of a global cost functional which combines a geometric prior on shape, a smoothness prior on radiance and a data fitness score. We estimate the shape and radiance via an alternating minimization: The radiance is computed as the solutions of partial differential equations defined on the surface and the background. The shape is estimated using a gradient descent flow, which is implemented using the level set method. Our algorithm works for scenes with smooth radiances as well as fine homogeneous textures, which are known challenges to traditional stereo algorithms based on local correspondence.

KEY WORDS: Variational methods; Mumford–Shah functional; image segmentation; multi-frame stereo reconstruction; partial differential equations; level set methods.

1. INTRODUCTION

Inferring the three-dimensional shape of a scene from a collection of images is one of the most studied problems in Computer Vision. In the case of multiple images, this problem is typically approached in two steps: first image points or regions are

[1] Department of Electrical Engineering, Washington University, Saint Louis, Missouri 63130. E-mail: hljin@ee.wustl.edu

[2] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332. E-mail: ayezzi@ece.gatech.edu

[3] PACM and Mathematics Department, Princeton University, Princeton, New Jersey 08544. E-mail: ytsai@math.princeton.edu. Research was performed while the author was with University of California, Los Angeles.

[4] Department of Mathematics, University of California San Diego, La Jolla, California 92093. E-mail: lcheng@math.ucsd.edu

[5] Computer Science Department, University of California, Los Angeles, California 90095. E-mail: soatto@cs.ucla.edu

0885-7474/03/1200-0267/0 © 2003 Plenum Publishing Corporation

matched across different images to establish local correspondences; then three-dimensional shape is inferred by combining the local correspondences.[6] Correspondence suffers significantly from the presence of noise and local minima, and therefore causes the overall system to break down.

One assumption which most matching algorithms rely on is that the radiance function of the scene is nowhere constant. In the case that the radiance is smooth or dominated by fine textures, image matching will give poor correspondence results. Therefore, we need to seek a global scheme in which we can leverage the prior on the radiance of the scene.

On the other hand, when the radiance of the scene is smooth, the appearance of its image projection will be smooth, except self-occlusions. Therefore, shape reconstruction corresponds to a well-defined problem in the image domain, that can be posed in terms of region-based segmentation [19].

In [27], Yezzi and Soatto proposed a method to solve the multi-frame shape reconstruction of smooth surfaces as the joint global region segmentation of the images. They assumed that the scene of interest was populated by a number of Lambertian surfaces with smooth or constant radiances as well as fine textures. They defined an energy functional which combined photometry, geometry and prior assumptions together and then formulated the problem in a variational framework. Although the authors of [27] presented a piecewise smooth model, their experimental evaluations were based on a simplified model in which the radiances of both the foreground and background were assumed to be constant. The implementation of this simpler piecewise constant model is considerably easier than the piecewise smooth case since the latter implementation (to be discussed in Sec. 3 that follows) must track functions on evolving manifolds, while the former case must only track evolving manifolds (and a few constants). In this paper, we will describe in detail the implementation techniques behind this more general piecewise smooth model and will demonstrate its greater flexibility and power in the experimental results at the end of the paper.

1.1. Relation to Previous Work

Since this work covers both the topics of segmentation and shape reconstruction, it relates to a vast body of literature. In stereo reconstruction (see [11] and references therein), one makes the assumption that the scene is Lambertian and the radiance is nowhere constant in order to establish local correspondence and then recovers a dense three-dimensional structure of the scene. Among all, there is a small body of literature on variational stereo [12, 14, 16], pioneered by Faugeras and Keriven's work [12]. They pose the problem in a variational framework, where the cost functional corresponds to a global matching score. In a sense, our work can be interpreted as extending the approach in [12] to regions, where the radiance is too smooth to establish local correspondences. In shape carving [18], the same

[6] There are exceptions to this scheme, for instance the variational approach to stereo, championed by Faugeras and Keriven [12], who combined the correspondence establishment and the shape reconstruction into one single step.

assumptions are used to recover a representation of shape (the *largest* shape that is photometrically consistent with the image data) as well as photometry. We use a different assumption, namely that the radiance and shape are smooth, to recover a different representation (the *smoothest* shape that is photometrically consistent with the data in a variational sense) as well as photometry. Therefore, our algorithm can be interpreted as performing space carving in a variational framework, minimizing the effect of noise. Note, however, that once a voxel is deleted by the carving procedure, it can never be retrieved. In this sense, shape carving is *uni-directional*. Our algorithm, on the other hand, is bi-directional, in that surfaces are allowed to evolve inward and outward. Finally, there is a relationship between our reconstruction method and the literature on shape from silhouettes [7], although the latter is based on local correspondence between occluding boundaries. In a sense, this work can be interpreted as a region-based method to reconstruct shape from silhouettes.

The material in this paper is tightly related to a wealth of contributions in the field of region-based segmentation, starting from Mumford and Shah's pioneering work [19], and including [6, 22, 28, 29]. This line of work stands to complement local edge-based segmentation methods such as [2, 3, 8, 15, 24, 25]. There are also algorithms that combine both features [4, 5].

In the direction of combining region-based segmentation and shape reconstruction, this paper continues the work of Yezzi and Soatto [27].

Since we would like to also give details about the numerical implementation, this paper has a close relation to the field of level set methods, introduced by Osher and Sethian [20].

1.2. Outline of This Paper

The rest of this paper is organized as follows: In Sec. 2, we will discuss the energy model proposed in [27] for joint image segmentation and shape reconstruction. We will formulate the problem in a variational framework and discuss a way to minimize the energy functional using partial differential equations (PDEs). In Sec. 3, we further elaborate on the implementation of these PDEs in the level set framework. We report our experimental results on synthetic datasets and real datasets in Sec. 4 and conclude this paper with a discussion of our algorithm in Sec. 5.

2. STEREOSCOPIC SEGMENTATION

We consider the problem of recovering the three-dimensional shape of a given scene from a set of two-dimensional images taken from multiple calibrated cameras. We assume that the scene is composed of a number of smooth, closed surfaces supporting smooth Lambertian radiance functions (or dense textures with spatially smooth statistics) and the background, which happens to occupy the entire field of view (the "blue sky" assumption) and supports a different radiance function. Under these assumptions, a subset of discontinuities (or texture discontinuities) in images correspond to occluding boundaries. These assumptions make the image segmentation problem well-posed, although not general. In fact,

"true" segmentation in this context corresponds in a one-to-one fashion to the shape of the surfaces in the scene. Based on these assumptions, one can set up a cost functional to minimize the variations within each image region, where the free parameters are not the boundaries in the image themselves, but the shape of the surfaces in space whose occluding contours happen to project onto such boundaries.

2.1. A Variational Framework

We consider S to be a smooth surface in $\mathbb{R}^3$ with generic local coordinates $(u, v) \in \mathbb{R}^2$. Let dA be its Euclidean area element, i.e., $dA = \|S_u \times S_v\| \, du \, dv$. The norm $\|\cdot\|$ we use here is the standard Euclidean norm. We denote with $\mathbf{X} = [X, Y, Z]^T$ the coordinates of a generic point on S with respect to a fixed inertial reference frame. We assume that we can measure n images: I_i, $i = 1, 2, \ldots, n$. All the images are fully calibrated, i.e., all the intrinsic and extrinsic parameters of the cameras are given. Let $\Omega_i \subset \mathbb{R}^2$ be the domain of the image I_i with area element $d\Omega_i$. We will use $\mathbf{X}_i = [X_i, Y_i, Z_i]^T$ to represent $\mathbf{X}$ in the "camera coordinates" with respect to the ith camera. The camera projection is modeled as an ideal perspective projection: $\pi_i : \mathbb{R}^3 \to \Omega_i; \mathbf{X} \mapsto \mathbf{x}_i$, where $\mathbf{x}_i = [x_i, y_i]^T = [X_i/Z_i, Y_i/Z_i]^T$. We describe the background B ("blue sky") as a sphere with angular coordinates $\Theta = (\theta, \eta) \in \mathbb{R}^2$ that may be related in a one-to-one manner with the coordinates $\mathbf{x}_i$ of each image domain Ω_i through the mapping Θ_i (i.e., $\Theta = \Theta_i(\mathbf{x}_i)$). We assume that the background supports a radiance function $\mathbf{g}: B \to \mathbb{R}$ and the surface supports another radiance function $\mathbf{f}: S \to \mathbb{R}$. We define the region R_i to be $\pi_i(S) \subset \Omega_i$ and denote its complement by R_i^c. Although the perspective projection π_i is not one-to-one (and therefore not invertible), the operation of back-projecting a point $\mathbf{x}_i$ from R_i onto the surface S still makes sense in the following way: For a given surface S, we trace the ray starting from the ith camera center and passing through $\mathbf{x}_i$, find its first intersection with S, and define the intersection point as the back-projection of $\mathbf{x}_i$ onto S. Therefore, we will make a slight abuse of notation and denote this back-projection by $\pi_i^{-1}: R_i \to S; \mathbf{x}_i \mapsto \mathbf{X}$. Please refer to Table I for quick references to the notations used in this paper.

In order to infer the shape of S, we impose a cost on the discrepancy between the projection of the model foreground and background radiances via the model surface and the actual measurements. Such a cost, E, depends upon the surface S as well as upon the radiances of the surface $\mathbf{f}$ and of the background $\mathbf{g}$: $E = E(\mathbf{f}, \mathbf{g}, S)$. We will then adjust the shape of the model surface and radiances to match the measured images. Since the unknowns (surface S and radiances $\mathbf{f}, \mathbf{g}$) live in infinite-dimensional spaces, we need to impose regularization. In particular, we can leverage on our assumption that the radiances are smooth. However, this is still not enough, for the estimated surface could converge to a very irregular shape to match image noise and fine details. Therefore, we impose a geometric prior on shape (smoothness). These are the three main ingredients in our approach: a data fidelity term $E_{\text{data}}(\mathbf{f}, \mathbf{g}, S)$ that measures the discrepancy between the measured images and the images predicted by the model, a smoothness term for the estimated radiances

Table I. Quick Reference to the Notation

Variables	Descriptions
S	the surface of interest
$(u, v) \in \mathbb{R}^2$	generic local coordinates for S
B	the background surface
$(\theta, \eta) \in \mathbb{R}^2$	the angular coordinates for B
$\mathbf{f}: S \to \mathbb{R}$	the radiance function of the surface S
$\mathbf{g}: B \to \mathbb{R}$	the radiance function of the background B
$\mathbf{X} = [X, Y, Z]^T$	the coordinates of a generic point on S
$\mathbf{X}_i = [X_i, Y_i, Z_i]^T$	$\mathbf{X}$ with respect to the reference frame of the ith camera
N	the inward unit normal to S
N_i	N with respect to the reference frame of the ith camera
Ω_i	the domain of the image I_i
$\mathbf{x}_i = [x_i, y_i]^T$	the coordinates of a point in Ω_i
$\pi_i: S \to \Omega_i$	the projection transformation of the ith camera
$\Theta_i: \Omega_i \to B$	the coordinate transformation from Ω_i to B
$R_i = \pi_i(S)$	the projection of the surface S in the ith image
$R_i^c = \Omega_i \setminus \pi_i(S)$	the projection of the background B in the ith image

$E_{\text{smooth}}(\mathbf{f}, \mathbf{g}, S)$ and a geometric prior $E_{\text{geom}}(S)$. We consider the composite cost functional to be a weighted average of these three terms:

$$E(\mathbf{f}, \mathbf{g}, S) = E_{\text{data}}(\mathbf{f}, \mathbf{g}, S) + \alpha E_{\text{geom}}(S) + \beta E_{\text{smooth}}(\mathbf{f}, \mathbf{g}, S). \tag{2.1}$$

where $\alpha, \beta \in \mathbb{R}^+$. In particular, the geometric prior is given by:

$$E_{\text{geom}} = \int_S dA = \text{area}(S), \tag{2.2}$$

while smoothness is imposed by a cost on the quadratic variations of the functions $\mathbf{f}$ and $\mathbf{g}$ on S:

$$E_{\text{smooth}} = \int_S \|\nabla_S \mathbf{f}\|^2 \, dA + \int_B \|\nabla_\Theta \mathbf{g}\|^2 \, d\Theta, \tag{2.3}$$

where ∇_S denotes the intrinsic gradient on the manifold S (the exact definition is given in Appendix A.) and ∇_Θ denotes the intrinsic gradient on B. Finally, the data fidelity term may be measured in the sense of $\mathscr{L}^2$ in the image domain by

$$E_{\text{data}} = \sum_{i=1}^{n} \int_{R_i} (\mathbf{f}(\pi_i^{-1}(\mathbf{x}_i)) - I_i(\mathbf{x}_i))^2 \, d\Omega_i + \sum_{i=1}^{n} \int_{R_i^c} (\mathbf{g}(\Theta_i(\mathbf{x}_i)) - I_i(\mathbf{x}_i))^2 \, d\Omega_i. \tag{2.4}$$

To facilitate the computation of the variation with respect to S, we express these integrals over the surface S. We start by introducing the characteristic functions $\chi_i(\mathbf{X})$ into the integrand. Here $\chi_i(\mathbf{X}) = 1$ if $\mathbf{X}$ is visible from the ith camera and $\chi_i(\mathbf{X}) = 0$ if $\mathbf{X}$ is not visible from the ith camera, i.e., $\mathbf{X}$ is occluded by other points on S with respect to the ith camera. We may express E_{data} as follows:

$$E_{\text{data}} = \sum_{i=1}^{n} \left(\int_{\Omega_i} \rho_i^2(\mathbf{x}_i)\, d\Omega_i + \int_{R_i} ((\mathbf{f}(\pi_i^{-1}(\mathbf{x}_i)) - I_i(\mathbf{x}_i))^2 - \rho_i^2(\mathbf{x}_i))\, d\Omega_i \right)$$

$$= \sum_{i=1}^{n} \left(\int_{\Omega_i} \rho_i^2(\mathbf{x}_i)\, d\Omega_i + \int_{\pi_i^{-1}(R_i)} (\tilde{\rho}_i^2(\mathbf{X}) - \rho_i^2(\pi_i(\mathbf{X})))\, \sigma_i(\mathbf{X}, N)\, dA \right)$$

$$= \sum_{i=1}^{n} \left(\int_{\Omega_i} \rho_i^2(\mathbf{x}_i)\, d\Omega_i + \int_{S} \chi_i(\mathbf{X})(\tilde{\rho}_i^2(\mathbf{X}) - \rho_i^2(\pi_i(\mathbf{X})))\, \sigma_i(\mathbf{X}, N)\, dA \right) \tag{2.5}$$

where $\tilde{\rho}_i(\mathbf{X}) = \mathbf{f}(\mathbf{X}) - I_i(\pi_i(\mathbf{X}))$ and $\rho_i(\mathbf{x}_i) = \mathbf{g}(\Theta_i(\mathbf{x}_i)) - I_i(\mathbf{x}_i)$. We use the fact that the area measure $d\Omega_i$ of the image is related to the area measure dA of the surface by $d\Omega_i = (\mathbf{X}_i \cdot N_i)/Z_i^3\, dA$, where N is the inward unit normal to S, and N_i is N with respect to the coordinates of the ith camera. $\sigma_i(\mathbf{X}, N)$ is a shorthand notation to $(\mathbf{X}_i \cdot N_i)/Z_i^3$.

Combining all three terms, we have the global cost functional as follows:

$$E(\mathbf{f}, \mathbf{g}, S) = \sum_{i=1}^{n} \left(\int_{\Omega_i} \rho_i^2(\mathbf{x}_i)\, d\Omega_i + \int_{S} \chi_i(\mathbf{X})(\tilde{\rho}_i^2(\mathbf{X}) - \rho_i^2(\pi_i(\mathbf{X})))\, \sigma_i(\mathbf{X}, N)\, dA \right)$$
$$+ \alpha \int_S dA + \beta \left(\int_S \|\nabla_S \mathbf{f}\|^2\, dA + \int_B \|\nabla \mathbf{g}\|^2\, d\Theta \right). \tag{2.6}$$

The model (2.6) was first proposed by Yezzi and Soatto in [27]. However, experimentally they only evaluated a simplified cost functional in which both radiance functions $\mathbf{f}$ and $\mathbf{g}$ were taken to be constant:

$$E_{\text{constant}} = E_{\text{data}} + E_{\text{geom}}$$
$$= \sum_{i=1}^{n} \left(\int_{\Omega_i} \varrho_i^2(\mathbf{x}_i)\, d\Omega_i + \int_{S} \chi_i(\mathbf{X})(\tilde{\varrho}_i^2(\mathbf{X}) - \varrho_i^2(\pi_i(\mathbf{X})))\, \sigma_i(\mathbf{X}, N)\, dA \right) + \alpha \int_S dA \tag{2.7}$$

where $\tilde{\varrho}_i(\mathbf{X}) = \mathbf{f} - I_i(\pi_i(\mathbf{X}))$ and $\varrho_i(\mathbf{x}_i) = \mathbf{g} - I_i(\mathbf{x}_i)$. $\mathbf{f}$ and $\mathbf{g}$ are constant over S and B respectively. This simplification corresponds to giving infinite weight to the smoothness term E_{smooth}, i.e., $\beta = \infty$. It is related to the approach of Chan and Vese in [6] who considered a piecewise constant version of the Mumford–Shah functional for image segmentation. In this paper, we consider the full energy model (2.6).

2.2. Gradient Descent Flows

We use an iterative procedure to find the surface S and the radiances $\mathbf{f}$ and $\mathbf{g}$ that minimize the functional (2.6). In particular, we start from an initial surface S, compute the optimal radiances $\mathbf{f}$ and $\mathbf{g}$, update S according to the gradient descent flow which is shown to minimize $E(\mathbf{f}, \mathbf{g}, S)$, and then repeat the whole procedure again. To fully carry out this procedure, we need to solve the following two problems: given $\mathbf{f}$ and $\mathbf{g}$, find the gradient descent flow for S; given S, compute the optimal $\mathbf{f}$ and $\mathbf{g}$.

In order to address the first problem, we need to find the gradient descent flow for each component of the energy with respect to the variation of S. The gradient descent flow for E_{geom} is simply the mean curvature flow, which minimizes the surface area:

$$S_t = 2HN, \tag{2.8}$$

where H is the mean curvature defined with respect to the inward normal N. We keep 2 in the flow to make sure the overall flow is correctly weighted with respect to the weights in the functional (2.6).

The flow to minimize E_{data} can be obtained by computing its first variation. The formula was derived in [27]:

$$S_t = \left(\sum_{i=1}^{n} \frac{1}{Z_i^3} ((\mathbf{f}-\mathbf{g})(\mathbf{f}+\mathbf{g}-2I_i)(\nabla_i \chi_i \cdot \mathbf{X}_i) + 2\chi_i (I_i - \mathbf{f})(\nabla_i \mathbf{f} \cdot \mathbf{X}_i)) \right) N, \tag{2.9}$$

where ∇_i denotes the intrinsic gradient with respect to $\mathbf{X}_i$ (recall that $\mathbf{X}_i$ is the representation of a point using the camera coordinates associated with the ith camera). We remark that the first term in the data flow involves the gradient of the characteristic function χ_i and is therefore non-zero only on the portions of S which project onto the boundary of the region R_i. As such, this term may be directly associated with a curve evolution equation for the boundary of the region R_i within the domain Ω_i of the image I_i. The second term, on the other hand, may be non-zero over the entire patch $\pi_i^{-1}(R_i)$ of S.

The evolution equation for the smoothness term needs some attention. First we notice that $\int_B \|\nabla_\Theta \mathbf{g}\|^2 \, d\Theta$ does not depend on S. Thus we only need to compute the variation of $\int_S \|\nabla_S \mathbf{f}\|^2 \, dA$ with respect to S. In the appendix, we show that the gradient descent flow for the energy $\int_S \|\nabla_S \mathbf{f}\|^2 \, dA$ is given by:

$$S_t = 2(\text{II}(\nabla_S \mathbf{f} \times N) - \|\nabla_S \mathbf{f}\|^2 H) \, N, \tag{2.10}$$

where $\text{II}(\mathbf{t})$ denotes the second fundamental form of a tangent vector $\mathbf{t}$ with respect to the inward normal. Note that $\nabla_S \mathbf{f} \times N$ is a tangent vector since it is perpendicular to N. The meaning of this flow becomes more clear when it is expressed in the level set framework (see Sec. 3.4). Essentially it is a combination a heat flow along one direction and a backward heat flow along another direction, both of which are in the tangent plane.

We may now write down the complete gradient flow for $E = E_{\text{data}} + \alpha E_{\text{geom}} + \beta E_{\text{smooth}}$ as

$$\begin{aligned} S_t = \Bigg(& \sum_{i=1}^{n} \frac{1}{Z_i^3} ((\mathbf{f}-\mathbf{g})(\mathbf{f}+\mathbf{g}-2I_i)(\nabla_i \chi_i \cdot \mathbf{X}_i) + 2\chi_i (I_i - \mathbf{f})(\nabla_i \mathbf{f} \cdot \mathbf{X}_i)) \\ & + 2\alpha H + 2\beta(\text{II}(\nabla_S \mathbf{f} \times N) - \|\nabla_S \mathbf{f}\|^2 H) \Bigg) N. \end{aligned} \tag{2.11}$$

A particularly nice feature of flow (2.11) (which is shared by the standard Mumford–Shah formulation for direct image segmentation) is that it depends only on the

image values, *not on the image gradients*. This factor makes the algorithm less sensitive to image noise when compared to other variational approaches for stereo (therefore less likely to cause the flow to become "trapped" in local minima).

2.3. Estimating the Scene Radiances

Once an estimate of the surface S is available, we need to find the optimal radiances $\mathbf{f}$ and $\mathbf{g}$ minimizing our energy functional $E(\mathbf{f}, \mathbf{g}, S)$. The minimizers $\mathbf{f}$ and $\mathbf{g}$ satisfy the Euler–Lagrange equations:

$$\beta \, \Delta_S \mathbf{f} = \sum_{i=1}^{n} \chi_i (\mathbf{f} - I_i) \, \sigma_i, \tag{2.12}$$

$$\beta \, \Delta_\Theta \mathbf{g} = \sum_{i=1}^{n} \hat{\chi}_i (\mathbf{g} - I_i), \tag{2.13}$$

where Δ_S denotes the Laplace–Beltrami operator on the surface S, Δ_Θ denotes the Laplacian on the background B with respect to its spherical coordinates Θ, and $\hat{\chi}_i(\Theta)$ denotes a characteristic function for the background B where $\hat{\chi}_i(\Theta) = 1$ if $\Theta_i^{-1}(\Theta) \in R_i^c$ and $\hat{\chi}_i(\Theta) = 0$ otherwise.

We solve these two elliptic PDEs by performing gradient descent using the following PDEs:

$$\frac{\partial \mathbf{f}}{\partial t} = \beta \, \Delta_S \mathbf{f} - \sum_{i=1}^{n} \chi_i (\mathbf{f} - I_i) \, \sigma_i, \tag{2.14}$$

$$\frac{\partial \mathbf{g}}{\partial t} = \beta \, \Delta_\Theta \mathbf{g} - \sum_{i=1}^{n} \hat{\chi}_i (\mathbf{g} - I_i). \tag{2.15}$$

The steady-state solutions to Eqs. (2.14), (2.15) will be the solutions of Eqs. (2.12), (2.13) respectively.

3. IMPLEMENTATION

The level set method is first introduced by Osher and Sethian in [20] as a numerical device to evolve interfaces. In the level set formulation, hyper-surfaces (for instance curves in two dimensions and surfaces in three dimensions) are represented implicitly as the zeros of a Lipschitz continuous function, often defined on a uniform Cartesian grid. With the robust numerical methods for Hamilton–Jacobi equations, see e.g., [10, 21, 23], the level set method handles topological changes of the surfaces during the evolution automatically, without performing any extra procedure (which is needed in explicit representations such as triangulations and splines). Furthermore, with the level set formulation, important geometrical quantities of the surfaces, such as the normals and curvatures, can be easily and accurately computed using finite difference schemes. These are the essential features needed for implementing Eqs. (2.11), (2.14), and (2.15).

3.1. Level Set Implementation

In this section we outline the level set implementation of flow (2.11). To simplify the notation, we may first re-write (2.11) as $S_t = \mathscr{F}N$, where $\mathscr{F}$ denotes a general speed function and N is the inward unit normal to S. In our case,

$$\mathscr{F} = \sum_{i=1}^{n} \frac{1}{Z_i^3}((\mathbf{f}-\mathbf{g})(\mathbf{f}+\mathbf{g}-2I_i)(\nabla_i \chi_i \cdot \mathbf{X}_i)+2\chi_i(I_i-\mathbf{f})(\nabla_i \mathbf{f}\cdot \mathbf{X}_i)) + 2\alpha H + 2\beta(\mathrm{II}(\nabla_S \mathbf{f}\times N)-\|\nabla_S \mathbf{f}\|^2 H). \tag{3.1}$$

The level set implementation of any geometric flow begins by embedding the initial interface $S(\mathbf{X}, 0)$ as the zero level set of a scalar function $\psi_0(\mathbf{X})$. We choose $\psi_0(\mathbf{X})$ to have the property that it is positive outside S, negative inside S and zero on S. $\psi_0(\mathbf{X})$ is then taken to be the initial condition for a function over time $\psi(\mathbf{X}, t)$:

$$\psi_0 \colon \mathbb{R}^3 \to \mathbb{R}, \qquad \psi \colon \mathbb{R}^3 \times \mathbb{R}^+ \to \mathbb{R}, \qquad \psi(\mathbf{X}, 0) = \psi_0(\mathbf{X}). \tag{3.2}$$

The key point is that we continue to embed the interface as the zero level set of ψ for all times. Thus, we have:

$$\psi_0(S(\mathbf{X}, 0)) = 0, \tag{3.3}$$

$$\psi(S(\mathbf{X}, t), t) = 0. \tag{3.4}$$

Differentiating (3.4) with respect to t therefore yields $\psi_t + \nabla\psi^T S_t = 0$, where $\nabla\psi$ is the gradient of ψ with respect to the spatial coordinates: $\nabla\psi = [\psi_x, \psi_y, \psi_z]^T$. Finally noting that the inward normal of S can be computed based on ψ as $N = -\nabla\psi/\|\nabla\psi\|$, we have

$$\psi_t = \mathscr{F}\|\nabla\psi\|. \tag{3.5}$$

3.2. Second Fundamental Form

To compute the speed function $\mathscr{F}$, we need to find a way to compute the second fundamental form for a given tangent vector in the level set framework. Expressing Eq. (3.4) with local coordinates (u, v), we have: $\psi(S(u, v), t) = 0$. Taking twice the derivatives with respect to u and v, we obtain:

$$\begin{cases} \psi_{uu} = S_u^T \nabla^2\psi S_u + \nabla\psi^T S_{uu} = 0 \\ \psi_{uv} = S_u^T \nabla^2\psi S_v + \nabla\psi^T S_{uv} = 0 \\ \psi_{vv} = S_v^T \nabla^2\psi S_v + \nabla\psi^T S_{vv} = 0 \end{cases}$$

where $\nabla^2\psi$ denotes the Hessian matrix of ψ with respect to the spatial coordinates:

$$\nabla^2\psi = \begin{bmatrix} \psi_{xx} & \psi_{xy} & \psi_{xz} \\ \psi_{yx} & \psi_{yy} & \psi_{yz} \\ \psi_{zx} & \psi_{zy} & \psi_{zz} \end{bmatrix}.$$

Now, recalling that the expression for the inward normal and that the coefficients e, f, g of the second fundamental form are given by $e = \langle N, S_{uu} \rangle$, $f = \langle N, S_{uv} \rangle$, and $g = \langle N, S_{vv} \rangle$, we obtain the following expressions:

$$e = \frac{1}{\|\nabla\psi\|} S_u^T \nabla^2\psi S_u, \qquad f = \frac{1}{\|\nabla\psi\|} S_u^T \nabla^2\psi S_v, \qquad g = \frac{1}{\|\nabla\psi\|} S_v^T \nabla^2\psi S_v.$$

Since S_u and S_v form a basis for the tangent space, we may conclude that the second fundamental form for any tangent vector $\mathbf{t}$ can be computed as:

$$\text{II}: T_{\mathbf{X}}S \to \mathbb{R} \qquad \mathbf{t} \mapsto \text{II}(\mathbf{t}) = \frac{1}{\|\nabla\psi\|} \mathbf{t}^T \nabla^2\psi \mathbf{t}, \tag{3.6}$$

where $T_{\mathbf{X}}S$ is the tangent plane of S at the point $\mathbf{X}$. We wish to make two remarks: First this form only makes sense for vectors in the tangent space. Second it is computed based upon the inward normal. In order to infer other second-order quantities of S, such as the mean and Gaussian curvatures, we can project the transformation (3.6) onto the space of 3×3 real symmetric matrices, and it becomes

$$\text{II} = \frac{1}{\|\nabla\psi\|} P_N^{\perp} \nabla^2\psi P_N^{\perp}, \tag{3.7}$$

where $P_V^{\perp}$ is the projection onto the orthogonal complement of the direction $V \in \mathbb{R}^3, V \neq 0$ and is given by:

$$P_V^{\perp} = I - \frac{VV^T}{\|V\|^2}. \tag{3.8}$$

As a 3×3 real symmetric matrix, II has the properties that 0 is one eigenvalue and N is the corresponding eigenvector. The principal curvatures (denoted as κ_1 and κ_2) are the other two eigenvalues, and the corresponding eigenvectors are the principal directions. Therefore, the mean curvature, defined as the average of the two principal curvatures, can be computed as follows:

$$H = \frac{\kappa_1 + \kappa_2}{2} = \frac{1}{2}\text{trace}(\text{II}) = \frac{1}{2}\nabla \cdot \left(\frac{\nabla\psi}{\|\nabla\psi\|} \right). \tag{3.9}$$

We remark that $\frac{1}{2}$ differentiates H from its usual definition in the level set framework, where the mean curvature is defined as the sum of the two principal curvatures.

Therefore, the flow (2.11) can be written in the level set framework as:

$$\begin{aligned} \psi_t = & \left(\sum_{i=1}^{n} \frac{1}{Z_i^3} ((\mathbf{f}-\mathbf{g})(\mathbf{f}+\mathbf{g}-2I_i)(\nabla_i \chi_i \cdot \mathbf{X}_i) + 2\chi_i (I_i - \mathbf{f})(\nabla_i \mathbf{f} \cdot \mathbf{X}_i)) \right) \|\nabla\psi\| \\ & + \frac{2\beta}{\|\nabla\psi\|^2} (\nabla_S \mathbf{f} \times \nabla\psi)^T \nabla^2\psi (\nabla_S \mathbf{f} \times \nabla\psi) - \beta \|\nabla_S \mathbf{f}\|^2 \nabla \cdot \left(\frac{\nabla\psi}{\|\nabla\psi\|} \right) \|\nabla\psi\| \\ & + \alpha \nabla \cdot \left(\frac{\nabla\psi}{\|\nabla\psi\|} \right) \|\nabla\psi\|. \end{aligned} \tag{3.10}$$

Note that Eq. (3.10) is a geometric PDE, i.e., it is homogeneous of degree one in $\nabla\psi$. This guarantees that our level set equation is invariant to the embedding function. We refer the reader to [9] for more theoretical aspects of this type of PDEs.

The numerical implementation of Eq. (3.10) goes as follows: The first term involving $\|\nabla\psi\|$ is discretized by Godunov's scheme for Hamilton–Jacobi Equations, see e.g., [20, 21] for details. The rest of the terms involving the spatial derivatives are discretized by standard compact central differencing. Time stepping is done by simple forward Euler method. Since Eq. (3.10) involves second order quantities, the CFL condition is $\Delta t = c_1 \Delta x^2$ where Δt is the time increment and Δx is the size of grid. c_1 will depend on the speed function.

3.3. Radiance Estimation

In addition to being able to evolve the surface S, we need a numerical device to track the optimal radiances $\mathbf{f}$ and $\mathbf{g}$ defined on S. This amounts to solving Eqs. (2.14) and (2.15) on a moving manifold. In [1], Bertalmio *et al.* proposed a framework for solving PDEs on static manifolds, which are represented by level set functions. Their idea is to implicitly represent the static surface using the level set function, and solve the equations defined on the surface in a fixed Cartesian coordinate system using this new embedding function. We shall use this framework to find the optimal $\mathbf{f}$ and $\mathbf{g}$ based on the current estimate of S. The overall algorithm will go in an alternating fashion: evolve S, estimate $\mathbf{f}$ and $\mathbf{g}$, and then repeat. Since both steps decrease the global cost functional, the algorithm converges.

For the current estimate of S, the method of finding the optimal radiances $\mathbf{f}$ and $\mathbf{g}$ goes in two steps: first extend the function, which is originally defined on the manifold, into the whole space[7]; second solve the original PDE in space, instead on the manifold. One possibility for doing the first step is to extend the function constant along the normal to the manifold. It is equivalent to solving the following PDE:

$$\nabla\mathbf{f}\cdot N = 0 \Leftrightarrow \nabla\mathbf{f}\cdot\nabla\psi = 0. \tag{3.11}$$

Recall that $\mathbf{f}$, the radiance function, is a scalar function defined on S. After adding a time variable, we can numerically seek the solution to Eq. (3.11) by iteratively searching for the steady state of the following PDE:

$$\frac{\partial\mathbf{f}}{\partial t} + \text{sign}(\psi)(\nabla\mathbf{f}\cdot\nabla\psi) = 0. \tag{3.12}$$

For the second step, Bertalmio *et al.* derived expressions for isotropic and anisotropic diffusions on manifolds. In the case of isotropic diffusion $\frac{\partial\mathbf{f}}{\partial t} = \Delta_S\mathbf{f}$, which is

[7] If one is using a narrow-band level set implementation, then this extension only needs to be done in the band where the computation occurs.

related with our problem, they obtained that the Laplace–Beltrami operator $\Delta_S \mathbf{f}$ can be computed using finite differencing as follows:

$$\Delta_S \mathbf{f} = \frac{1}{\|\nabla\psi\|} \nabla \cdot (P^{\perp}_{\nabla\psi} \nabla \mathbf{f} \|\nabla\psi\|). \tag{3.13}$$

Equation (3.13) can be further simplified as follows:

$$\begin{aligned}
\Delta_S \mathbf{f} &= \frac{1}{\|\nabla\psi\|} \nabla \cdot (P^{\perp}_{\nabla\psi} \nabla \mathbf{f} \|\nabla\psi\|) \\
&= \frac{1}{\|\nabla\psi\|} \nabla \cdot \left(\nabla \mathbf{f} \|\nabla\psi\| - \nabla \mathbf{f}^T \nabla\psi \frac{\nabla\psi}{\|\nabla\psi\|} \right) \\
&= \Delta \mathbf{f} + \frac{\nabla \mathbf{f}^T \nabla^2 \psi \nabla\psi}{\|\nabla\psi\|^2} - \nabla \left(\frac{\nabla\psi}{\|\nabla\psi\|} \right) \frac{\nabla \mathbf{f}^T \nabla\psi}{\|\nabla\psi\|} - \frac{\nabla\psi^T \nabla^2 \mathbf{f} \nabla\psi}{\|\nabla\psi\|^2} - \frac{\nabla \mathbf{f}^T \nabla^2 \psi \nabla\psi}{\|\nabla\psi\|^2} \\
&= \Delta \mathbf{f} - \nabla \left(\frac{\nabla\psi}{\|\nabla\psi\|} \right) \frac{\nabla \mathbf{f}^T \nabla\psi}{\|\nabla\psi\|} - \frac{\nabla\psi^T \nabla^2 \mathbf{f} \nabla\psi}{\|\nabla\psi\|^2} \\
&= \nabla \cdot (P^{\perp}_{\nabla\psi} \nabla \mathbf{f})
\end{aligned} \tag{3.14}$$

where $\nabla^2 \mathbf{f}$ is the Hessian matrix of $\mathbf{f}$ with respect to the spatial coordinates. $\nabla \cdot (P^{\perp}_{\nabla\psi} \nabla \mathbf{f})$ is the expression for the Laplace–Beltrami operator that Bertalmio *et al.* found when the surface S is defined by its signed distance function. Indeed it is true for any embedding function and leads to a simpler finite difference scheme for implementation.

We may now implement Eq. (2.14) as

$$\frac{\partial \mathbf{f}}{\partial t} = \beta \left(\Delta \mathbf{f} - \nabla \left(\frac{\nabla\psi}{\|\nabla\psi\|} \right) \frac{\nabla \mathbf{f}^T \nabla\psi}{\|\nabla\psi\|} - \frac{\nabla\psi^T \nabla^2 \mathbf{f} \nabla\psi}{\|\nabla\psi\|^2} \right) + \sum_{i=1}^{n} \chi_i (\mathbf{f} - I_i) \sigma_i. \tag{3.15}$$

Equation (2.15) can be implemented similarly. Observing the simple expression for $\Delta_S \mathbf{f}$ we use compact central differencing on all the spatial derivatives of $\mathbf{f}$ and ψ, and forward Euler for time. The CFL condition for Eq. (3.15) is that $\Delta t = c_2 \Delta x^2$ and c_2 will depend on the curvature of the manifold S and β.

3.4. Tuning of Parameters

Note that $\nabla_S \mathbf{f}$, $\nabla_S \mathbf{f} \times N$, and N are orthogonal to each other. If we define $\nabla^{\perp}_S \mathbf{f}$ to be $\nabla_S \mathbf{f} \times N$, $\overline{\nabla^{\perp}_S \mathbf{f}}$ to be $\frac{\nabla^{\perp}_S \mathbf{f}}{\|\nabla^{\perp}_S \mathbf{f}\|}$, and $\overline{\nabla_S \mathbf{f}}$ to be $\frac{\nabla_S \mathbf{f}}{\|\nabla_S \mathbf{f}\|}$, then $\overline{\nabla^{\perp}_S \mathbf{f}}$, $\overline{\nabla_S \mathbf{f}}$ and N form an orthonormal frame for $\mathbb{R}^3$. We can thus re-write the terms regarding to the smoothness energy in Eq. (3.10) as:

$$\frac{2\beta}{\|\nabla\psi\|^2}(\nabla_S\mathbf{f}\times\nabla\psi)^T\nabla^2\psi\,\nabla_S\mathbf{f}\times\nabla\psi-\beta\|\nabla_S\mathbf{f}\|^2\nabla\cdot\left(\frac{\nabla\psi}{\|\nabla\psi\|}\right)\|\nabla\psi\|$$

$$=2\beta\|\nabla_S\mathbf{f}\|^2\frac{\partial^2\psi}{\partial^2\overline{\nabla_S^\perp\mathbf{f}}}-\beta\|\nabla_S\mathbf{f}\|^2\left(\Delta\mathbf{f}-\frac{\nabla\psi^T\nabla^2\psi\,\nabla\psi}{\|\nabla\psi\|^2}\right)$$

$$=\beta\|\nabla_S\mathbf{f}\|^2\left(2\frac{\partial^2\psi}{\partial^2\overline{\nabla_S^\perp\mathbf{f}}}-\left(\frac{\partial^2\psi}{\partial^2\overline{\nabla_S^\perp\mathbf{f}}}+\frac{\partial^2\psi}{\partial^2\overline{\nabla_S\mathbf{f}}}+\frac{\partial^2\psi}{\partial^2N}-\frac{\partial^2\psi}{\partial^2N}\right)\right)$$

$$=\beta\|\nabla_S\mathbf{f}\|^2\left(\frac{\partial^2\psi}{\partial^2\overline{\nabla_S^\perp\mathbf{f}}}-\frac{\partial^2\psi}{\partial^2\overline{\nabla_S\mathbf{f}}}\right). \tag{3.16}$$

The flow for the smoothness term is a combination of a heat flow along the direction of $\nabla_S\mathbf{f}\times N$ and a backward heat flow along the direction of $\nabla_S\mathbf{f}$. Therefore, this flow is not stable in the latter direction. However, we can make the flow for the overall cost functional stable, by overweighting the unstable part from the geometric part. The requirement is:

$$\alpha\geqslant\beta\max_S\|\nabla_S\mathbf{f}\|^2. \tag{3.17}$$

Equation (3.17) is the stability condition. We may choose α and β based on this condition. Since β controls the allowed variations for both radiances, we should choose β with respect to the variations present in the scene. For large variations, we shall use small β and for small variations, we shall use large β. Once β is chosen, α is chosen to satisfy (3.17).

3.5. Visibility

At each step, in order to compute the speed function, we need to compute χ_i. It is equivalent to the problem of determining which part of the surface is visible from a given view point (the camera center in our case). This is a classical problem in Computer Graphics. A typical approach to this problem is the so-called *ray tracing*. The idea is to start from each point in the domain of interest, shoot a ray towards the view point, and check the number of times this ray hits the surface. Unfortunately this intuitive algorithm turns out to be computationally expensive. Let N be the number of points needed to resolve a one dimensional object. The overall complexity of this approach is then $\mathcal{O}(N^4)$. Indeed, it is possible to solve the visibility problem efficiently in the level set framework.

In the rest of this section, we describe an implementation of the implicit ray tracing technique that is originally reported in [26]. This is a one-pass algorithm that finds the line of sight for a given configuration of implicit surfaces in an incremental way. In our implementation described below, the complexity is $\mathcal{O}(N^3)$ with a very small constant. However, it can be made much faster by skipping through large regions using multi-resolution techniques [26].

We assume that our surface S is defined as the zeros of the level set function ψ in a closed domain Ω: $S\subset\Omega\subset\mathbb{R}^3$ and the position of the view point is $\mathbf{v}$. We are

going to compute another level set function ϕ, which tells us the portions of Ω that are visible from the view point. More precisely, $\Omega^+ \doteq \{\mathbf{X} \in \Omega : \phi(\mathbf{X}) \geqslant 0\}$ will be the regions visible from $\mathbf{v}$, and the desired characteristic function can be written as the composition of the Heaviside function and ϕ: $\chi = H(\phi)$. Of course, all of these functions and quantities are defined on a uniform Cartesian grid in a rectangular domain Ω_d. We will use the conventional indexing scheme to enumerate the grid nodes in Ω; i.e., $\mathbf{X}_{i,j,k} = \mathbf{X}_{ll} + \Delta x(i, j, k)$, where $\mathbf{X}_{ll}$ denotes the lower left corner of Ω_d, and Δx denotes the mesh size. With this notation at hand, we define a voxel to be the cube with vertices $\{\mathbf{X}_{i+p,j+q,k+r}: p, q, r \in \{0, 1\}\} \subset \Omega_d$ for some (i, j, k).

A key part of this visibility algorithm is to organize all the points in Ω in a special order so that the causality of sight is not violated. This is done by ordering the grid points into the queue Q described in the Appendix of [26]. We will describe this queue later in this section.

At each grid point $\mathbf{X}$ in the queue, we first find the voxel containing $\mathbf{X}$. From this voxel, we find the face F that intersects the line segment L joining $\mathbf{v}$ and $\mathbf{X}$. (Here we think of L as an open line segment, not containing either $\mathbf{v}$ or $\mathbf{X}$.) This face is called the *upwind* face. If L intersects F at a point, we call this intersection $\mathbf{X}'$. If L lies in F, the problem is reduced to a two dimensional problem, and $\mathbf{X}'$ is the intersection of L and an edge of F.

In the case where L intersects F at a point, we perform a bilinear interpolation to approximate the value of ϕ at $\mathbf{X}'$: $\phi(\mathbf{X}')$. Note that this interpolation is feasible because the values of ϕ on all the neighboring grid points around $\mathbf{X}'$ (including $\mathbf{X}'$ if $\mathbf{X}'$ happens to be a grid point as well) have been computed, due to the special ordering.

Finally, we update the value of $\phi(\mathbf{X})$ by:

$$\phi(\mathbf{X}) = \min(\psi(\mathbf{X}), \phi(\mathbf{X}')). \tag{3.18}$$

Thus we are passing visibility information along the ray emanating from the view point and the minimum is applied to fix the position of the obstructions. Note that if $\mathbf{X}$ satisfies $\psi(\mathbf{X}) < 0$, then $\phi(\mathbf{X})$ will be less than zero in the end, meaning $\mathbf{X}$ is not visible. So even if $\phi(\mathbf{X}')$ is positive, it will not make $\phi(\mathbf{X})$ positive.

As mentioned above, the interpolation procedure is made possible by the special ordering of the grid points. We now provide one example of such ordering for completeness of our exposition. In the following pseudo-code, (v_1, v_2, v_3) denotes the grid index of the view point $\mathbf{v}$, and `Q` is the queue of interest. `m` is the index of `Q`. The grid is bounded by 0 and `nx` in the X direction, 0 and `ny` in the Y direction and 0 and `nz` in the Z direction.

```
m=0;
for(s1=-1;s1<=1;s1+=2)
  for(s2=-1;s2<=1;s2+=2)
    for(s3=-1;s3<=1;s3+=2)
      for(i=v1;(s1<0?i>=0:i<=nx);i+=s1)
        for(j=v2;(s2<0?j>=0:j<=ny);j+=s2)
          for(k=v3;(s3<0?k>=0:k<=nz);k+=s3)
            Q[m++]=(i,j,k);
```

This is called the sweeping sequence in [26]. Using multiple directions (here parameterized by `s1`, `s2`, and `s3`) of sweeping is simply to propagate visibility information in all directions of the rays emanating from the visible point as quickly as possible. If the sweeping direction were always the same, then information would propagate quickly over rays pointing generally in that direction, however, other directions would suffer. Thus we use several sweeping directions to cover all the possible directions of the rays.

The algorithm thus reads:

```
for (each X in the queue Q) do
  Find the upwind face F and X' ∈ F;
  Compute φ(X');
  Update φ(X) := min(ψ(X), φ(X'));
end
```

We direct readers to [26] for more details on the implicit ray tracing.

3.6. Projecting ε-Neighborhoods of the Tangent Plane

Given a point $\mathbf{X}_0$ on the surface, we need to determine its projection on the image and then get the intensity value at that point. Since interpolation is necessary to obtain the intensity value, one has to know the interpolation region. The problem of determining the size of the interpolation region is directly related to the overall robustness of the algorithm to image noise. Imagine that the three-dimensional grid one uses is very coarse, and he/she also chooses a small interpolation region. Then the resulting algorithm will be very sensitive to image noise.

The way we approach the problem is to start from the image domain. We pose the following question: given the point $\mathbf{X}_0$ with the projection $\mathbf{x}_0$, and $\epsilon > 0$ what is the neighborhood of $\mathbf{x}_0$ whose back-projections onto S are within the ε-neighborhood of $\mathbf{X}_0$. We answer this question by approximating the surface S locally around $\mathbf{X}_0$ using the tangent plane passing through $\mathbf{X}_0$. The tangent plane satisfies the following equation:

$$N \cdot \mathbf{X} = N \cdot \mathbf{X}_0. \tag{3.19}$$

Let (u, v) be the projection of $\mathbf{X}$ in image domain, the tangent plane may then be parameterized via the image coordinates (u, v) as follows:

$$\mathbf{X}(u, v) = \frac{N \cdot \mathbf{X}_0}{N \cdot (u, v, 1)} (u, v, 1). \tag{3.20}$$

Let (u_0, v_0) be the projection of $\mathbf{X}_0$, then we may write

$$(u, v, 1) = \frac{\mathbf{X}_0}{Z_0} + \Delta \qquad \text{where} \quad \Delta = (u - u_0, v - v_0, 0)$$

which, when plugged into (3.20), gives the following expression for the tangent plane in terms of Δ:

$$\mathbf{X}(u, v) = \frac{N \cdot \mathbf{X}_0}{N \cdot (\mathbf{X}_0 + Z_0 \Delta)} (\mathbf{X}_0 + Z_0 \Delta). \tag{3.21}$$

Now, based upon this local approximation of S by its tangent plane through $\mathbf{X}_0$, we may test whether the back-projection of a point (u, v) in a neighborhood of (u_0, v_0) is within an ϵ-neighborhood of $\mathbf{X}_0$ by checking if $\|\mathbf{X}(u, v) - \mathbf{X}_0\| < \epsilon$.

4. EXPERIMENTS

In this section, we report the experimental results of the proposed algorithm conducted on synthetic and real datasets. In Fig. 1 we show 4 out of 24 views of a synthetic scene that contains two spheres as the foreground. Both spheres have smooth radiances ranging from dark to white and the background has a constant radiance, whose value (gray) is right in the middle of the radiances of the spheres. This scene would represent a big challenge to traditional correspondence-based stereo algorithms, because the radiances are too smooth. It also cannot be solved by

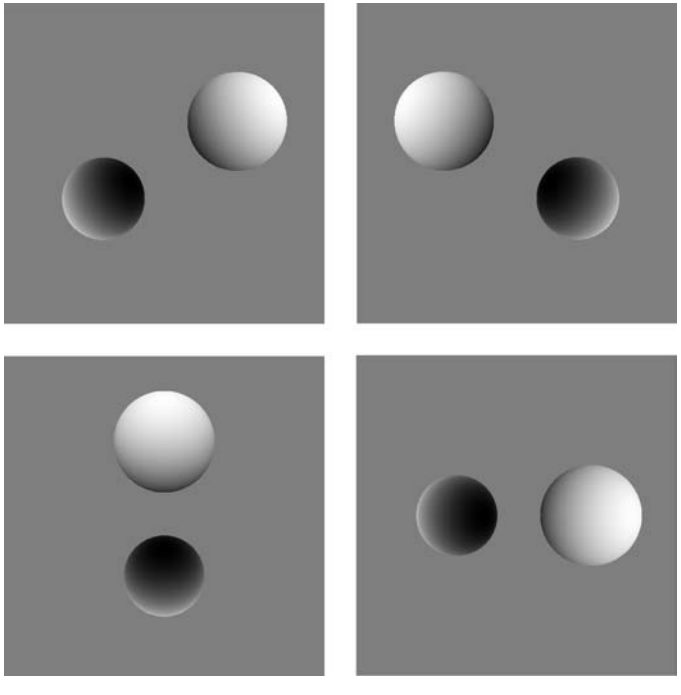

Fig. 1. The two-sphere dataset (4 out of 24 views). Both spheres have smooth radiances ranging from dark to white, and the background has a constant radiance, whose value (gray) is right in the middle of the radiances of the spheres.

Fig. 2. The final reconstructed surface obtained with the algorithm based on the model (2.7). The algorithm captures the dark parts of the scene. This corresponds to a foreground radiance and a background radiance with the largest difference.

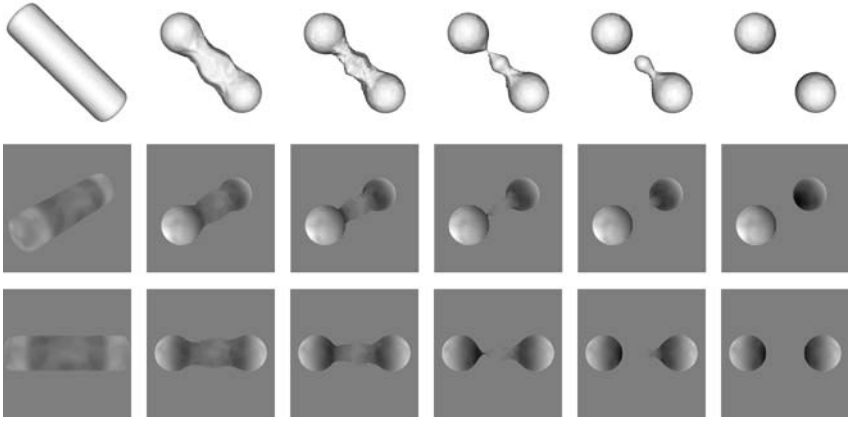

Fig. 3. Top row: rendered surface evolving from a long bar to the final solid model. Bottom two rows: the modeled images of the scene at the corresponding times from two selected views. They are obtained by projecting the current estimated radiances of the surfaces and the background onto the selected view.

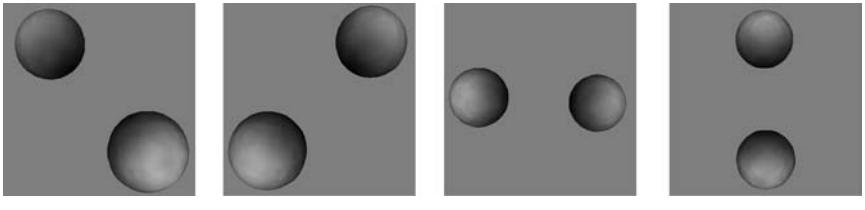

Fig. 4. Selected views of the final reconstructed surface. These images are not obtained using texture-mapping from one or more images in the original dataset. They are obtained using the final estimation of the radiance functions for the surface and the background. The intensity value of the background is set to be 127. (The image intensities range from 0 to 255.) Note that even some parts of the radiances of both spheres are the same as that of the background. The final reconstructed models are still round and smooth. This is due to the geometric prior on the shape and smoothness prior on the radiance.

approximating both radiances as constants, because it violates completely the constant radiance assumption. This is verified by our experimental evaluation. For comparison purpose, we first run the algorithm based on the simplified model (2.7) on this dataset. In Fig. 2 we show the final reconstructed surface. The initial surface is a long bar that contains both spheres. Due to the simplification in the model, the algorithm with constancy assumption can only capture the dark parts of the scene, which corresponds to the largest difference between the foreground radiance and the background radiance. Then we run the algorithm based on the full model (2.6) on the same dataset. In Fig. 3 we show the surface evolving from the same initial surface to a final solid model. In Fig. 4 we show the final reconstructed surface from various vantage points. We remark that these images are not obtained by texture-mapping using one or more images from the original dataset. Instead they are obtained by projecting the estimated radiance functions $\mathbf{f}$ and $\mathbf{g}$ onto the selected view. Note that even some parts of the radiances of both spheres are the same of the radiance of the background. The algorithm can still keep the shape round and smooth, thanks to the geometric prior on the shape and the smoothness prior on the radiance.

In Fig. 5 we show 4 out of 16 views of another synthetic scene whose foreground is composed of five objects: two spheres, one cube, one cylinder, and one cone. The data is obtained as follows: we first build a graphical model of the scene and then render all the views using OpenGL. All five objects have distinct albedos,

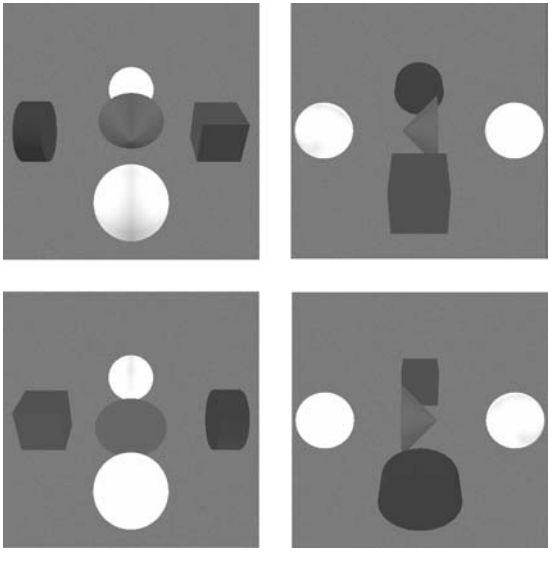

Fig. 5. 4 out of 16 views of a scene, which contains five objects: two spheres, one cube, one cylinder and one cone. They all have distinct albedos, but constant within each object. However due the light condition, the radiances of all the objects are not constant, but smooth. The background is corrupted with white Gaussian noise.

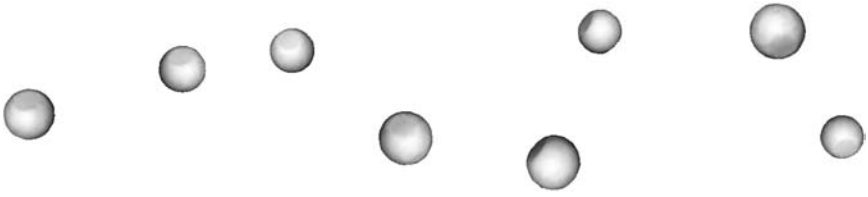

Fig. 6. The final reconstructed surface obtained with the algorithm based on the model (2.7). The algorithm captures the two white spheres where the radiances of the foreground and background have the largest difference.

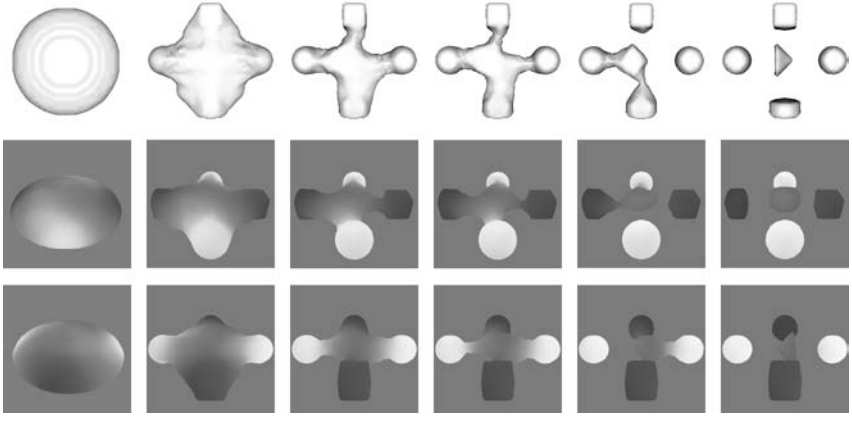

Fig. 7. Top row: the rendered surface during evolution. Notice that that the initial surface neither contains nor is contained by the final surfaces. Bottom two rows: the modeled images of the scene at the corresponding times from two selected views. They are obtained by projecting the current estimation of the radiances of the surfaces and the background onto the selected views.

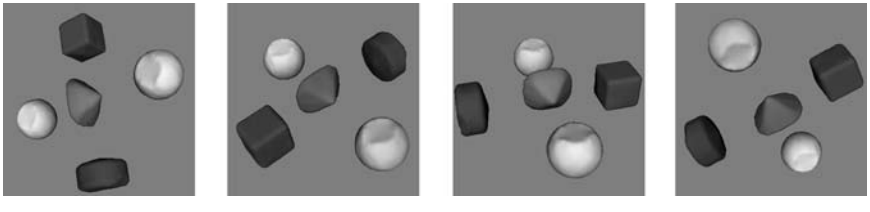

Fig. 8. Selected views of the final reconstructed surface. The surface is rendered using the final estimation of the radiance function. The intensity value of the background is set to 127. Notice that two sides of the both spheres are flattened. That is due to the insufficient data and the geometric prior: first the data is ambiguous for round spheres and flattened spheres; second the geometric prior will give a unique result with the minimal surface area, which is the flattened spheres.

Fig. 9. The "watering can" sequence and the initial surface. In order to capture the surface, it is necessary that the initial surface intersects with the true surface. One way to guarantee this is to start with a number of small surfaces, two in this case.

but constant within each object. However, due to the lighting condition (there are two point light sources in the scene), the radiance of each object is not constant, but smooth. The background is corrupted with white Gaussian noise. In Fig. 6 we show the results of the algorithm based on the simplified model (2.7). The initial surface shape is an ellipsoid that neither contains nor is contained by the scene objects. This algorithm only captures the two spheres which also corresponds to the largest difference between the foreground and background radiances. In Figs. 7 and 8 we show the results obtained by running the algorithm based on the full energy model (2.6). Figure 7 contains rendered views of the surface evolution and the modeled images (obtained using the estimated radiance functions) from two selected viewpoints. Figure 8 contains the final reconstructed surface rendered from a few vantage points. These images are, again, rendered by projecting the estimated radiances.

In Fig. 9 we show an image from a dataset which contains 31 views of a watering can, together with the initial surface. Due to the lighting condition, the watering can exhibits variations in the radiance. In particular, the bottom part is darker than the top part. Observing that the radiance is nearly constant, we start the estimation process from the model (2.7) and then switch to our full model (2.6) in the end. In Fig. 10 we show the rendered surface evolving from two ellipsoids to the final model. The final estimated shape is shown Fig. 11 from a few viewpoints.

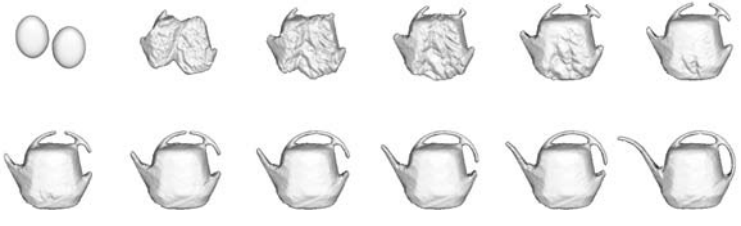

Fig. 10. Rendered surface during evolution for the watering can.

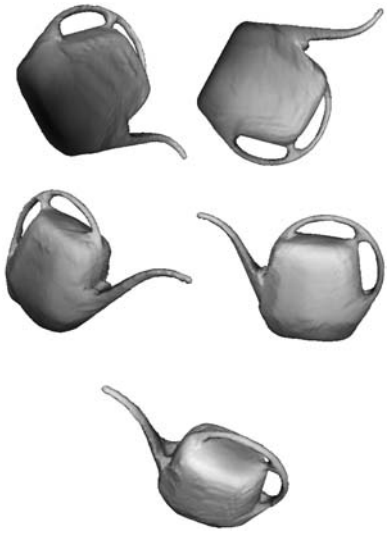

Fig. 11. The final estimated surface shown from several viewpoints. The images are obtained with the final estimation of the radiance functions of the watering can and the background. One can see that the bottom part of the watering can is darker than the top. It is due to the shading caused by the lighting condition in the original data. Although the base is not visible from any camera, our algorithm can give a unique radiance, which is smooth and consistent with the rest part of the watering can. Note that in order to perceive better the difference in the radiance, we have linearly mapped the radiance from 80–180 to 0–255.

In Fig. 12 we show 4 out of 33 images from another dataset which contains a statue of a skater. The radiance value of the background is within the range of the radiance value of the statue. In particular, the top head of the skater is brighter than the background, while some other parts are darker. The model (2.7) fails to capture shape of the statue, because the assumptions are violated. The results are reported in Fig. 13. We show the results based on the model (2.6) in Fig. 14. The final estimated shape is shown in Fig. 15 from a few viewpoints.

5. DISCUSSION

We have presented an iterative algorithm to estimate the shape and radiance of a Lambertian scene from a number of calibrated images. The radiance of both foreground and background are assumed to be smooth functions and the shape is also smooth. We formulate the problem in a variational framework where we estimate all the unknowns (shape and radiances) by minimizing a global cost functional. This algorithm is an extension of the one proposed by Yezzi and Soatto [27], in which they only consider constant radiance. We discuss the detailed implementation, which has been carried out in the level set framework.

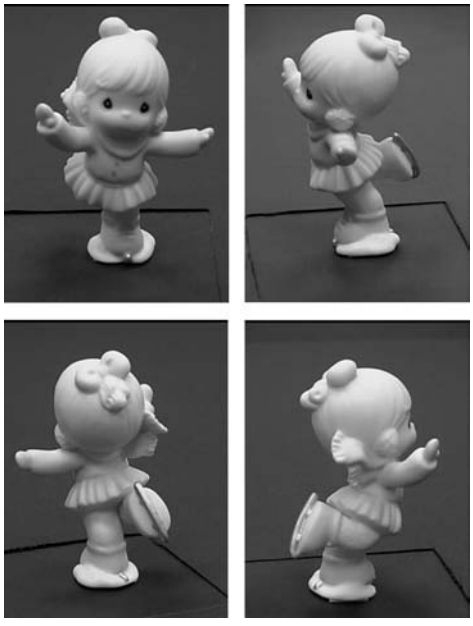

Fig. 12. 4 out of total 31 views of a statue of skater.

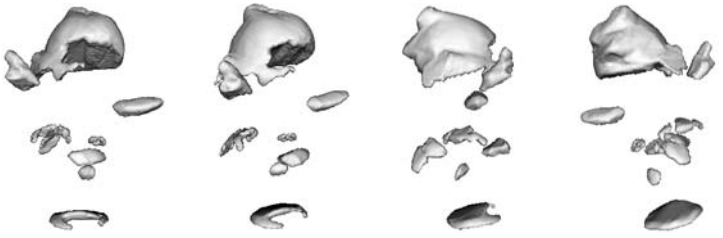

Fig. 13. The final reconstructed surface obtained with the algorithm based on the model (2.7).

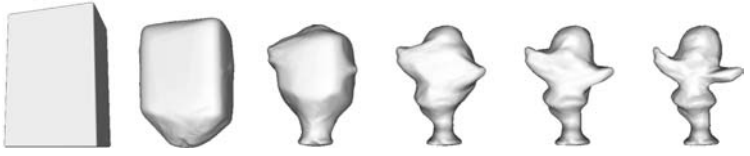

Fig. 14. Rendered surfaces during evolution.

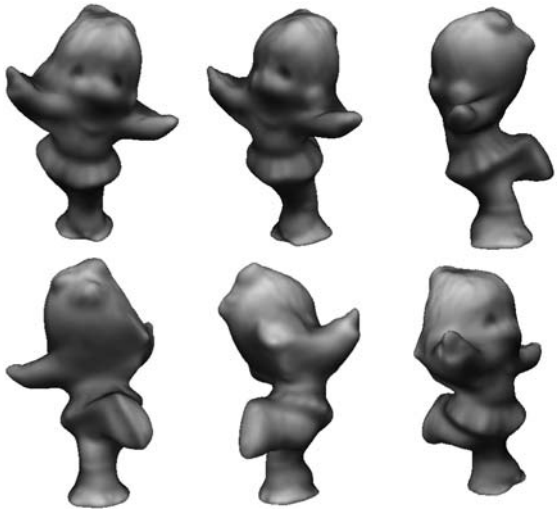

Fig. 15. The final estimated surface shown from several viewpoints. The statue is "textured" with the final estimated radiance and then fed into a shading renderer. Note that in order to perceive better the difference in the radiance, we have linearly mapped the radiance from 50–235 to 0–255.

APPENDIX A. GRADIENT DESCENT FLOW FOR $\int_S \|\nabla_S \mathbf{f}\|^2 \, dA$

To the end of computing the gradient descent flow, we must first express this integral using a fixed parameterization (u, v), which is independent of S itself. We will further assume in this computation that S may be covered by a single coordinate patch. This seems like a very restrictive assumption to start, but in the end, the resulting PDE will have a completely local expression which does not depend on this global assumption. Throughout our computation, we will always omit purely tangential terms whenever they appear inside an inner product with the shape variation S_t, because they will not change the shape of S. And we use T_* to denote any tangential term.

Let M be the matrix of the first fundamental form coefficients, i.e.,

$$M = \begin{bmatrix} E & F \\ F & G \end{bmatrix} \qquad (E = S_u \cdot S_u,\ F = S_u \cdot S_v,\ G = S_v \cdot S_v)$$

with respect to the local coordinates (u, v). $\nabla_S \mathbf{f}$ is the intrinsic gradient of $\mathbf{f}$ on the manifold S defined as follows:

$$\nabla_S \mathbf{f} = [S_u, S_v]\, M^{-1} \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix} \tag{A.1}$$

and

$$\begin{aligned}\|\nabla_S \mathbf{f}\|^2 &= \left\langle [S_u, S_v]\, M^{-1} \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix}, [S_u, S_v]\, M^{-1} \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix} \right\rangle \\ &= \left\langle \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix}, M^{-1}[S_u, S_v]^T [S_u, S_v]\, M^{-1} \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix} \right\rangle \\ &= \left\langle \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix}, M^{-1} \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix} \right\rangle = \frac{\mathbf{f}_u^2 G - 2\mathbf{f}_u \mathbf{f}_v F + \mathbf{f}_v^2 E}{EG - F^2} \end{aligned} \tag{A.2}$$

$\|\nabla_S \mathbf{f}\|^2$ is so-called the first differential parameter of Beltrami [17]. We can now rewrite the integral expression for $\int_S \|\nabla_S \mathbf{f}\|^2\, dA$ as

$$\int_S \|\nabla_S \mathbf{f}\|^2\, dA = \iint \frac{\mathbf{f}_u^2 G - 2\mathbf{f}_u \mathbf{f}_v F + \mathbf{f}_v^2 E}{\sqrt{EG - F^2}}\, du\, dv.$$

Note that for the sake of simplicity, we have omitted the integration domain for u and v. The first variation of the integral with respect to time can be computed as follows:

$$\begin{aligned}\frac{\partial}{\partial t} \int_S \|\nabla_S \mathbf{f}\|^2\, dA &= \iint \frac{\partial}{\partial t} \left(\frac{\mathbf{f}_u^2 G - 2\mathbf{f}_u \mathbf{f}_v F + \mathbf{f}_v^2 E}{\sqrt{EG - F^2}} \right) du\, dv \\ &= \iint \left(\frac{\mathbf{f}_u^2 G_t - 2\mathbf{f}_u \mathbf{f}_v F_t + \mathbf{f}_v^2 E_t}{\sqrt{EG - F^2}} \right. \\ &\quad \left. - \frac{1}{2} \frac{\mathbf{f}_u^2 G - 2\mathbf{f}_u \mathbf{f}_v F + \mathbf{f}_v^2 E}{(\sqrt{EG - F^2})^3} (E_t G + E G_t - 2F F_t) \right) du\, dv \\ &= \iint \left(\frac{\mathbf{f}_u^2 G_t - 2\mathbf{f}_u \mathbf{f}_v F_t + \mathbf{f}_v^2 E_t}{\sqrt{EG - F^2}} - \frac{1}{2} \|\nabla_s \mathbf{f}\|^2 \frac{E_t G + E G_t - 2F F_t}{\sqrt{EG - F^2}} \right) du\, dv \\ &= \iint \left(\frac{2\mathbf{f}_u^2 \langle S_v, S_{vt} \rangle - 2\mathbf{f}_u \mathbf{f}_v (\langle S_u, S_{vt} \rangle + \langle S_v, S_{ut} \rangle) + 2\mathbf{f}_v^2 \langle S_u, S_{ut} \rangle}{\sqrt{EG - F^2}} \right. \\ &\quad \left. + \|\nabla_s \mathbf{f}\|^2 \langle 2HN + T_*, S_t \rangle \right) du\, dv \\ &= -2 \iint \left(\frac{\mathbf{f}_u^2 \langle S_{vv}, S_t \rangle - 2\mathbf{f}_u \mathbf{f}_v \langle S_{uv}, S_t \rangle + \mathbf{f}_v^2 \langle S_{uu}, S_t \rangle}{\sqrt{EG - F^2}} \right. \\ &\quad \left. + \|\nabla_s \mathbf{f}\|^2 \langle 2HN, S_t \rangle + \langle T_*, S_t \rangle \right) du\, dv \\ &= -2 \iint \left\langle S_t, \frac{\mathbf{f}_u^2 g - 2\mathbf{f}_u \mathbf{f}_v f + \mathbf{f}_v^2 e}{EG - F^2} - \|\nabla_s \mathbf{f}\|^2 HN + T_* \right\rangle \sqrt{EG - F^2}\, du\, dv \\ &= -2 \int_S \left\langle S_t, \left(\frac{\mathbf{f}_u^2 g - 2\mathbf{f}_u \mathbf{f}_v f + \mathbf{f}_v^2 e}{EG - F^2} - \|\nabla_s \mathbf{f}\|^2 H \right) N \right\rangle dA. \end{aligned}$$

Recall that e, f, and g are the second fundamental form coefficients for the coordinates (u, v). If we let

$$S_t = 2\left(\frac{\mathbf{f}_u^2 g - 2\mathbf{f}_u\mathbf{f}_v f + \mathbf{f}_v^2 e}{EG - F^2} - \|\nabla_s \mathbf{f}\|^2 H\right) N, \tag{A.3}$$

$\frac{\partial}{\partial t}\int_S \|\nabla_S \mathbf{f}\|^2 \, dA \leqslant 0$. Therefore (A.3) minimizes $\int_S \|\nabla_S \mathbf{f}\|^2 \, dA$.

We may further simplify the first term: $\frac{\mathbf{f}_u^2 g - 2\mathbf{f}_u\mathbf{f}_v f + \mathbf{f}_v^2 e}{EG - F^2}$, by noting the following:

$$\begin{aligned}
\nabla_S \mathbf{f} \times N &= \left([S_u, S_v]\, M^{-1} \begin{bmatrix} \mathbf{f}_u \\ \mathbf{f}_v \end{bmatrix}\right) \times N \\
&= \frac{G\mathbf{f}_u - F\mathbf{f}_v}{EG - F^2}(S_u \times N) + \frac{-F\mathbf{f}_u + E\mathbf{f}_v}{EG - F^2}(S_v \times N) \\
&= \frac{1}{EG - F^2}\left((G\mathbf{f}_u - F\mathbf{f}_v)[S_u, S_v]\, M^{-1} \begin{bmatrix} 0 \\ \sqrt{EG - F^2} \end{bmatrix}\right. \\
&\quad \left. + (-F\mathbf{f}_u + E\mathbf{f}_v)[S_u, S_v]\, M^{-1} \begin{bmatrix} -\sqrt{EG - F^2} \\ 0 \end{bmatrix}\right) \\
&= \frac{1}{\sqrt{EG - F^2}}[S_u, S_v]\, M^{-1} \begin{bmatrix} F\mathbf{f}_u - E\mathbf{f}_v \\ G\mathbf{f}_u - F\mathbf{f}_v \end{bmatrix} \\
&= \frac{1}{\sqrt{EG - F^2}}[S_u, S_v]\, M^{-1} M \begin{bmatrix} -\mathbf{f}_v \\ \mathbf{f}_u \end{bmatrix} \\
&= [S_u, S_v] \frac{1}{\sqrt{EG - F^2}} \begin{bmatrix} -\mathbf{f}_v \\ \mathbf{f}_u \end{bmatrix}.
\end{aligned} \tag{A.4}$$

Therefore the second fundamental form of $\nabla_S \mathbf{f} \times N$:

$$\mathrm{II}(\nabla_S \mathbf{f} \times N) = \frac{1}{\sqrt{EG - F^2}}[-\mathbf{f}_v \quad \mathbf{f}_u] \begin{bmatrix} e & f \\ f & g \end{bmatrix} \frac{1}{\sqrt{EG - F^2}} \begin{bmatrix} -\mathbf{f}_v \\ \mathbf{f}_u \end{bmatrix} = \frac{\mathbf{f}_u^2 g - 2\mathbf{f}_u\mathbf{f}_v f + \mathbf{f}_v^2 e}{EG - F^2}.$$

We may thus re-write (A.3) as:

$$S_t = 2(\mathrm{II}(\nabla_S \mathbf{f} \times N) - \|\nabla_S \mathbf{f}\|^2 H)\, N. \tag{A.5}$$

ACKNOWLEDGMENTS

This research is supported in part by NSF Grants IIS-9876145, IIS-0208197, CCR-0133736, DMS-0112413, DMS-0208449, and DMS-0074735, ARO Grant DAAD19-99-1-0139, Intel Grant 8029, ONR Grant N00014-97-1-0270, and NIH Grant 1-P20-MH-65166-1. We wish to thank two anonymous reviewers for their suggestions.

REFERENCES

1. Bertalmio, M., Cheng, L., Osher, S., and Sapiro, G. (2001). Variational problems and partial differential equations on implicit surfaces. *J. Comput. Phys.* **174**, 759–780.

2. Caselles, V., Catte, F., Coll, T., and Dibos, F. (1993). A geometric model for active contours in image processing. *Numer. Math.* **66**, 1–31.
3. Caselles, V., Kimmel, R., and Sapiro, G. (1997). Geodesic active contours. *Int. J. Comput. Vision* **22**, 61–79.
4. Chakraborty, A., and Duncan, J. (1999). Game-Theoretic Integration for image segmentation. *IEEE Trans. Pattern Anal. Machine Intell.* **21**, 12–30.
5. Chakraborty, A., Staib, L., and Duncan, J. (1996). Deformable boundary finding in medical images by integrating gradient and region information. *IEEE Trans. Medical Imaging* **15**, 859–870.
6. Chan, T., and Vese, L. (2001). Active contours without edges. *IEEE Trans. on Image Processing* **10**, 266–277.
7. Cipolla, R., and Blake, A. (1992). Surface shape from the deformation of apparent contours. *Int. J. Comput. Vision* **9**.
8. Cohen, L. (1991). On active contour models and balloons. *CVGIP: Image Understanding* **53**, 211–218.
9. Crandall, M. G., Ishii, H., and Lions, P. L. (1992). User's guide to viscosity solutions of second order partial differential equations. *Bull. Amer. Math. Soc. (N.S.)* **27**, 1–67.
10. Crandall, M. G., and Lions, P. L. (1984). Two approximations of solutions of Hamilton–Jacobi equations. *Math. Comp.* **43**, 1–19.
11. Faugeras, O. (1993). *Three Dimensional Vision, a Geometric Viewpoint*, MIT Press.
12. Faugeras, O., and Keriven, R. (1998). Variational principles, surface evolution pdes, level set methods and the stereo problem. *IEEE Trans. Image Process.* **7**, 336–344.
13. Horn, B., and Brooks, M. (eds.) (1989). *Shape from Shading*, MIT Press.
14. Jin, H., Yezzi, A., and Soatto, S. (2000). Stereoscopic shading: Integrating shape cues in a variational framework. *Proc. of the IEEE Intl. Conf. on Computer Vision and Pattern Recognition*, pp. 169–176.
15. Kass, M., Witkin, A., and Terzopoulos, D. (1987). Snakes: Active contour models. *Int. J. Comput. Vision* **1**, 321–331.
16. Kimmel, R. (2002). 3D shape reconstruction from autostereograms and stereo. *J. Vis. Commun. Image Repres.* **13**, 324–333.
17. Kreyszig, E. (1991). *Differential Geometry*, Dover.
18. Kutulakos, K., and Seitz, S. (2000). A theory of shape by space carving. *Int. J. Comput. Vision* **38**, 199–218.
19. Mumford, D., and Shah, J. (1989). Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. Pure Appl. Math.* **42**, 577–685.
20. Osher, S., and Sethian, J. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi equations. *J. Comput. Phys.* **79**, 12–49.
21. Osher, S., and Shu, C.-W. (1991). High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations. *SIAM J. Numer. Anal.* **28**, 907–922.
22. Ronfard, R. (1994). Region-based strategies for active contour models. *Int. J. Comput. Vision* **13**, 229–251.
23. Souganidis, P. E. (1985). Approximation schemes for viscosity solutions of Hamilton–Jacobi equations. *J. Differential Equations* **59**, 1–43.
24. Tek, H., and Kimia, B. (1995). Image segmentation by reaction diffusion bubbles. *Proc. Int. Conf. Computer Vision*, pp. 156–162.
25. Terzopoulos, D., and Witkin, A. (1988). Constraints on deformable models: Recovering shape and non-rigid motion. *Artificial Intelligence* **36**, 91–123.
26. Tsai, Y.-H., Cheng, L.-T., Burchard, P., Osher, S., and Sapiro, G. (2002). *Dynamic Visibility in an Implicit Framework*, UCLA CAM Report, 02-06.
27. Yezzi, A., and Soatto, S. (2001). Stereoscopic segmentation. *Proc. Int. Conf. Computer Vision*, 56–66.
28. Yezzi, A., Tsai, A., and Willsky, A. (1999). A statistical approach to image segmentation for bimodal and trimodal imagery. *Proc. Int. Conf. Computer Vision*, 898–903.
29. Zhu, S., and Yuille, A. (1996). Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Trans. Pattern Anal. Machine Intelligence* **18**, 884–900.