Modeling of Free-Form Surfaces and Shape From Shading

Subhajit Sanyal Mayank Bansal Subhashis Banerjee Prem K Kalra
Department of Computer Science & Engineering
Indian Institute of Technology Delhi
Hauz Khas, New Delhi, 110016, India
{subhajit, mayban, suban, pkalra}@cse.iitd.ernet.in

Abstract

Modeling a free-form 3D-surface from a single view has been a widely pursued problem. The existing schemes are either fully-automatic shape-from-X techniques or involve adept interaction from the user but little or no geometric (photometric) basis. We propose a novel scheme of interactive modeling of free-form lambertian surfaces where the solution obtained is consistent with the Shape from Shading model. To this end, a reinforcement learning based scheme has been adopted which allows user intervention at any stage of the algorithm to guide the SFS solution to a global minimum.

1. Introduction

Recovering 3D shape and geometry information from a single image has been an area of active interest in the area of computer vision. Depending on the type of visual cues utilized there is a whole gamut of techniques which address this challenging issue. Techniques which address structured scenarios utilize various geometric invariants [2]. Methods applicable for free-form shapes utilize shading [5], texture [1], contour information [13], etc. as cues. User interaction/intervention in varying degrees is a necessity in a vast majority of such single-view techniques. Methods dealing with structured scenes require fair amount of user interaction in the identification of the geometric invariants. Techniques addressing free-form surface reconstruction typically involve the user in the specification of seed solutions, boundary conditions and in the identification of discontinuities, folds, creases [14] etc.

In this paper we present a semi-automatic scheme for single-view modeling of free-form lambertian surfaces. The technique in this contribution builds upon the purely interactive surface modeling scheme of *Zhang et al.* [14] by utilizing shading information to considerably reduce the user interaction in situations where the scene is known to satisfy certain shading constraints. In *Pictorial Relief* [7] Koenderink investigates the depth perception abilities in human visual system, and this forms the motivation for the work in [14]. However, apart from the visual appearance, little or nothing can be said about the reconstructed surface in terms of consistency with geometric (photometric) properties when using such a purely interactive scheme. If such a scheme is boot-strapped with constraints from the scene then the task of modeling becomes considerably easier, and this has been the motivation behind this contribution. We utilize *shape from shading* (SFS) constraints to decide the depths at user-specified control points which ultimately decide the shape of the piece-wise continuous patches. The survey [15] contains detailed study of several approaches to solve the SFS problem. A basic objection to a SFS formulation might be the issue of non-uniqueness and the fact that majority of the algorithms dealing with SFS can at most reach a local minimum as a solution. Also most of these algorithms do not allow much scope for the user to steer it to an acceptable solution. To address these issues we have proposed a novel reinforcement learning based scheme for solving the SFS problem which facilitates user intervention at any stage to guide the algorithm to a desired result. A useful feature of our scheme is the incremental nature of the modeling phase. Already reconstructed patches are utilized to provide boundary conditions to patches under construction. In this aspect the scheme is different from that in [14] where the minimization of the energy functional has to be done over all the constraints every time a new constraint is added to the set of existing ones. While majority of the SFS techniques deal with light sources at infinity, our formulation efficiently incorporates light sources placed at finite distances from the object. Moreover our SFS scheme, like the recent works of *Prados et al.* [8], also allows a more naturally occurring perspective camera as the imaging device in contrast to the majority of SFS techniques which limit themselves to the orthographic case.

The remainder of the paper is organized as follows. Section 2 poses the SFS problem in a variational setting which

Proceedings of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'04)
0-7695-2223-8/04 $ 20.00 IEEE

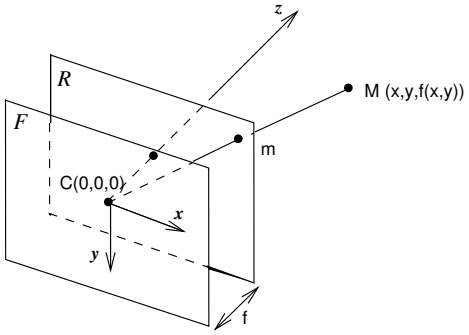


Figure 1. The Camera Coordinate System

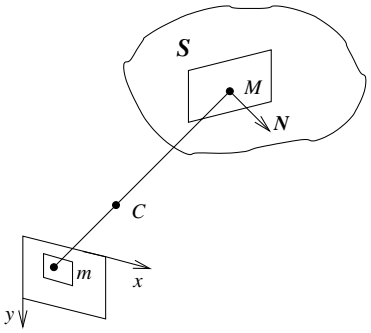


Figure 2. In the image, a square window around the projected pixel m is considered

naturally incorporates regularization in the error metric. To minimize the error we have utilized a stochastic reinforcement learning algorithm and its basic notion is presented in section 3. The algorithm is outlined in section 4. Section 5 illustrates the user interaction scheme with an example. The scheme of hierarchical reconstruction for dealing with complex surfaces is presented in section 6. Section 7 deals with the finer aspects of the technique, section 8 presents the results and section 9 concludes the contribution.

2. Problem Formulation

We represent the object to be modeled as a piecewise continuous function, $f(x,y)$ [14] which represents the *depth map*. The objective is to determine the value of the function f at some fixed (chosen by the user) locations (x,y) such that a continuous smooth surface fitted through these set of 3D points represents the actual surface as closely as possible. We propose to use the *Shape from Shading* (SFS) framework by utilizing the shading information from the given image, as a measure of the degree of match between the actual and the reconstructed surfaces. The existence and uniqueness of solution being an important aspect in the SFS problem, we propose an interactive setting (with learning automata as the underlying basis) where the user can pull the system out of local minima and guide it to an acceptable solution.

In what follows, we describe a *variational* formulation (similar to the formulation for stereo in [3]) for the shape from shading problem that incorporates *perspective projection* and *single point light source - finite or infinite*.

We will assume here that the camera performs a perspective projection of the 3D world on the retinal plane as shown in Fig.1. The optical center, noted C in the figure, is the center of projection and also the origin of our world coordinate system. The z-axis is directed towards the object and is perpendicular to the retinal plane. The image of the 3D point M is the pixel m on the retinal plane $\mathcal{R}$. Such a transformation can be described by a perspective projection matrix $\mathbf{P}$. Assuming that the focal length f of the camera is known, $\mathbf{P}$ takes the simple form:

$$\mathbf{P} = \mathbf{K}\left[\begin{array}{cc} \mathbf{I_3} & \mathbf{0} \end{array}\right]$$

where $\mathbf{K}$ is the camera internal matrix [4].

We denote the 3D surface by $\mathcal{S}$ which can be conveniently represented as $(x,y,f(x,y))$ in the coordinate system of the camera focal plane, described above. Let M be a point on this surface and $\mathbf{N}$ be the surface normal at M as depicted in Fig.2. Let ρ denote the (known) surface albedo and $\mathbf{l}$ denote the (again known) light source location in the coordinate system described above. The intensity value in the image at pixel m is denoted by $I(m)$ or $I(u,v)$.

We now define our *variational error metric*. The surface $\mathcal{S}$ represents a photometrically consistent solution for the *lambertian* shading equation [6] when the following mean square error is minimized for each point $M = (x,y,f(x,y))^T$ on the surface:

$$E(\mathbf{S},\mathbf{N},x,y) = \frac{1}{4pq}\int_{-p}^{p}\int_{-q}^{q}(I(m+m') - \rho\frac{\mathbf{L}.\mathbf{N}}{|\,\mathbf{L}\,||\,\mathbf{N}\,|})^2\,\mathrm{d}m' \tag{1}$$

where,

$$\mathbf{L} = \mathbf{l} - (x,y,f(x,y))^T$$

is the local light source direction at the point M

$$\begin{array}{rrcl} & \mathbf{N} & = & (f_x, f_y, -1)^T \\ \text{and,} & m & \equiv & \mathbf{P}.(x,y,f(x,y))^T \end{array}$$

The total error can then be written as:

$$C(\mathbf{S},\mathbf{N}) = \iint E(\mathbf{S},\mathbf{N},x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{2}$$

In (1), the integration is carried over a small window of size $4pq$ in the image assuming that the corresponding (small) patch on the surface $\mathcal{S}$ has the same normal $\mathbf{N}$ and the same local light source direction $\mathbf{L}$. This approach automatically imposes a smoothness constraint on the surface and regularizes the variational problem. The integral in (2) is carried over the entire surface $\mathcal{S}$ to obtain the total cost metric.

Next, we detail the important aspects of the learning automata framework.

3. Continuous Action Learning Automaton (CALA)

Learning automata are adaptive decision making units that learn to choose an optimal action from a set actions by interacting with a random environment. A continuous action learning automaton is an automaton with a continuous space of actions, i,e, the set of actions of a CALA is the real line $\mathcal{R}$. Details of Continuous Action Learning Automaton schemes can be found in [9],[11]. For completeness we present a brief overview. Associated with a CALA is a normal probability distribution $N(\mu(k),\sigma(k))$ which dictates the action chosen by the CALA at any instant of time k. This distribution is usually referred to as the action probability distribution. The CALA updates the mean $\mu(k)$ and standard deviation $\sigma(k)$ based on the feedback it receives from the environment. For an action x taken by the CALA, the reinforcement β_x provided by the environment is a random variable with an associated probability distribution. The expected feedback or the expected reinforcement received from the environment for the chosen action x is denoted as d_x, where

$$d_x = E[\beta_x|x] \tag{3}$$

The objective of the CALA is to learn the value of x for which d_x is maximum.

To describe the operations of a CALA, we consider the following situation where the task is to maximize a function $f : \mathcal{R} \rightarrow \mathcal{R}$. Given a value of x the function value $f(x)$ is given to the CALA as the feedback for its action. Thus $E[\beta_x|x] = f(x)$. The optimal action x_0 is the one which maximizes $f(x)$. To arrive at this value x_0 the CALA starts with an initial action probability distribution $N(\mu(0),\sigma(0))$ and, through a learning algorithm, updates its action probability distribution after each interaction with the environment. The CALA operates in the following fashion. At every time-step k, it selects a real number $x(k)$ at random guided by its current probability distribution $N(\mu(k),\phi(\sigma(k)))$, where the function $\phi(\sigma(k))$ is defined in equation (6). The CALA then obtains reinforcement from the environment for two actions, $\mu(k)$ and $x(k)$. Let these responses be denoted by $\beta_{\mu(k)}$ and $\beta_{x(k)}$ respectively. Based on these responses the CALA modifies its action probability distribution $N(\mu(k),\sigma(k))$ according to the following equations (4) and (5).

$$\mu(k+1) = \mu(k) + \lambda \frac{\left(\beta_{x(k)} - \beta_{\mu(k)}\right)}{\phi(\sigma(k))} \frac{\left(x(k) - \mu(k)\right)}{\phi(\sigma(k))} \tag{4}$$

$$\begin{aligned} \sigma(k+1) = \sigma(k) & \\ + \quad & \lambda \frac{\left(\beta_{x(k)} - \beta_{\mu(k)}\right)}{\phi(\sigma(k))} \left[\left(\frac{x(k) - \mu(k)}{\phi(\sigma(k))} \right)^2 - 1 \right] \\ - \quad & \lambda K(\sigma(k) - \sigma_L) \end{aligned} \tag{5}$$

where,

$$\begin{aligned} \phi(\sigma) &= \sigma_L \text{ for } \quad \sigma \leq \sigma_L \\ &= \sigma \quad \text{for} \quad \sigma > \sigma_L > 0 \end{aligned} \tag{6}$$

and λ is the learning parameter controlling the step size $(0 < \lambda < 1)$. K is a large positive number controlling the shrinking of σ and σ_L gives a lower bound for σ. The algorithm is said to have converged when $\mu(k)$ stops changing *appreciably* and $\sigma(k)$ is close to σ_L. The rationale behind the updation schemes presented above is the following. If $x(k)$ receives a better response compared to $\mu(k)$, then $\mu(k)$ is shifted towards $x(k)$ otherwise it is moved away. The standard deviation $\sigma(k)$ is shrunk normally, apart from situations when an action choice $x(k)$ *far away* from $\mu(k)$ receives a better response or when a $x(k)$ *very close* to $\mu(k)$ receives a worse response.

The asymptotic behavior of the CALA algorithm has been explored in [11], where the general algorithm presented above has been shown to be equivalent to a stochastic difference equation. With some assumptions which essentially specify the smoothness of the function $f(x)$ the algorithm, in the asymptotic sense, has a stochastic gradient following property. If σ and λ are sufficiently small, it can be shown that the algorithm converges to a close approximation of an isolated local maximum of $f(x)$.

3.1. Games of Continuous Action Learning Automata

Let us consider a situation where we want to maximize a function $f : \mathcal{R}^N \rightarrow \mathcal{R}$, where we call the arguments of f as parameters. To achieve this objective we consider a stochastic game of N Continuous Action Learning Automata [11], [10] corresponding to the N parameters of the function f. Each round of the game consists of each of the players choosing a specific action from their respective action sets. The CALA then receive payoffs for their selected actions

from the environment. The payoffs are the function values evaluated at the specific values of the actions selected by the N CALAs. In game playing terminology such a situation is called a *common payoff* game.

Let $\langle \mathbf{x}(k) \rangle$ denote the tuple of actions chosen by the N CALA at time-step k. In response to the tuple of actions chosen, the environment provides the stochastic reinforcement β to each of the CALA. We assume that β takes values in $[0,1]$. Let

$$F(\mathbf{x}) = E[\beta | j^{th} \text{ CALA chooses } x_j \in \mathcal{R}] \tag{7}$$

An action tuple $\langle \mathbf{x}^* \rangle$ is said to be *optimal* if

$$F(\mathbf{x}^*) \geq F(\mathbf{x}) \ \forall \mathbf{x} \in \mathcal{B}^N(\mathbf{x}^*, \varepsilon) \tag{8}$$

Here $\mathcal{B}^N(\mathbf{x}^*, \varepsilon)$ is an ε-ball in $\mathcal{R}^N$ centered at $\mathbf{x}^*$. Thus if any one of the players chooses a different strategy while the others keep their strategy unchanged the payoff decreases and the optimal action set is a Nash equilibrium. In other words $\mathbf{x}^*$ is a local maximum of $F(\mathbf{x})$.

The algorithm for a network of CALA in a game playing situation is similar to the one described above for a single CALA. Each CALA C_j in the network is associated with a normal action probability distribution $N(\mu_j(k), \sigma_j(k))$, where k is the time-step index as before. At the k^{th} time-step, the environment is presented with the action tuples $\langle x_1(k), x_2(k), x_3(k), \ldots, x_j(k), \ldots, x_N(k) \rangle$ and $\langle \mu_1(k), \mu_2(k), \mu_3(k), \ldots, \mu_j(k), \ldots, \mu_N(k) \rangle$. The environment computes the responses $\beta_{\langle \mathbf{x}(k) \rangle}$ and $\beta_{\langle \mu(k) \rangle}$ for the two tuples respectively. Then all the automata update their action probability distributions in the following fashion

$$\begin{aligned} \mu_j(k+1) &= \mu_j(k) \\ &+ \lambda \mathcal{F}_1(\mu_j(k), \sigma_j(k), x_j(k), \beta_{\langle \mathbf{x}(k) \rangle}, \beta_{\langle \mu(k) \rangle}) \end{aligned} \tag{9}$$

$$\begin{aligned} \sigma_j(k+1) &= \sigma_j(k) \\ &+ \lambda \mathcal{F}_2(\mu_j(k), \sigma_j(k), x_j(k), \beta_{\langle \mathbf{x}(k) \rangle}, \beta_{\langle \mu(k) \rangle}) \\ &- \lambda K[\sigma_j(k) - \sigma_L] \end{aligned} \tag{10}$$

where the functions $\mathcal{F}_1$ and $\mathcal{F}_2$ are defined as follows.

$$\mathcal{F}_1(\mu, \sigma, x, \beta_{\langle \mathbf{x} \rangle}, \beta_{\langle \mu \rangle}) = \left(\frac{\beta_{\langle \mathbf{x} \rangle} - \beta_{\langle \mu \rangle}}{\phi(\sigma)} \right) \left(\frac{x - \mu}{\phi(\sigma)} \right) \tag{11}$$

$$\mathcal{F}_1(\mu, \sigma, x, \beta_{\langle \mathbf{x} \rangle}, \beta_{\langle \mu \rangle}) = \left(\frac{\beta_{\langle \mathbf{x} \rangle} - \beta_{\langle \mu \rangle}}{\phi(\sigma)} \right) \left[\left(\frac{x - \mu}{\phi(\sigma)} \right)^2 - 1 \right] \tag{12}$$

Here $\phi(\sigma)$ is the same function defined in equation (6) and K and λ carry the same implications as before.

In [11], [10], one can find a detailed study of the asymptotic behavior of the algorithm. For games with *common payoff*, the algorithm converges to one of the maximal points of the function f, if certain assumptions, essentially regarding the smoothness of the response function, hold true. One can argue that the response function in our scheme as described in (14) satisfies these assumptions.

We note that the extremely useful feature of the above algorithm is that it does not require any gradient information. The algorithm stochastically seeks the optimum and follows the gradient in an expected sense and converges to the optimal points. However, such an algorithm might turn out to be slow in practice. This shortcoming is suitably answered in [12] where the authors show that the algorithm has immense potential for parallelizability. In fact utilizing n computing resources, one can obtain n-fold speed-up.

4. The Basic Surface Reconstruction Algorithm

In the image space, a set of points $\mathcal{B}_I$, on the boundary (visible silhouette) of the object is specified. Subsequently, some locations inside this boundary $\mathcal{C}_I$ (whose purpose will become clear in the ensuing discussion) are also indicated. The region inside the boundary is specified as a mask and is used to determine the pixels in the image which will be used in the *shading* computation. As an example, in Fig.6, the points in $\mathcal{B}_I$ are shown as empty circles while the points in $\mathcal{C}_I$ are shown as filled circles. The white region in Fig.6 is the primary region of interest (mask) where the depth computation is performed. The object is assumed to be at some *average depth* Z_{avg} from the camera. Let, $\mathcal{B}_W$ and $\mathcal{C}_W$ denote sets of world points (on the plane $z = Z_{avg}$) corresponding to those in the sets $\mathcal{B}_I$ and $\mathcal{C}_I$ respectively, obtained from a knowledge of the camera. All the points are thus associated with this depth i.e. $f(x,y) = Z_{avg}, \ \forall (x,y) \in \mathcal{B}_W \cup \mathcal{C}_W$. While the boundary points are kept fixed, the depths of the interior points keep changing as the algorithm proceeds. In Fig.3 we see a snap-shot of a surface at a certain time step of the algorithm where the positions of the points in $\mathcal{C}_W$ are indicated by black dots.

Now, the objective of the algorithm is to determine the values $f(x,y)$ for all points $(x,y) \in \mathcal{C}_W$ such that the piecewise continuous function fitted through these 3D points represents the photometrically consistent shape of the object. As introduced in section 3, this problem can be formulated as a stochastic game between CALA. We associate a CALA $\mathcal{C}_i$ with each of the locations (x_i, y_i) in $\mathcal{C}_W$. The action variable for each CALA $\mathcal{C}_i$ is the depth $f(x_i, y_i)$ of the surface at the point (x_i, y_i). We propose a game with *common payoff* with the variational error metric introduced in section 2 acting as the common reinforcement.

At any stage of the algorithm, the reinforcement β_z pro-

vided to the CALA for the chosen action set $\langle z_i \rangle$ is computed as follows (it is to be noted that the reinforcement β_μ for the mean set $\langle \mu_i \rangle$ is also computed in an identical fashion and henceforth we refer to the reinforcement simply as β unless specified otherwise). First, a (continuous) smooth surface $\mathcal{S}$ is fitted through the points $\{(x_i, y_i, z_i) : (x_i, y_i) \in \mathcal{C}_W\} \cup \{(x, y, Z_{avg}) : (x, y) \in \mathcal{B}_W\}$ by minimizing a *thin plate functional* similar to that used in [14]. Now, for evaluation of the variational error metric, we discretize the integrals in (2) to obtain:

$$C(\mathbf{S}, \mathbf{N}) = \sum_{x_g} \sum_{y_g} E(\mathbf{S}, \mathbf{N}(x_g, y_g, z_g), x_g, y_g) \qquad (13)$$

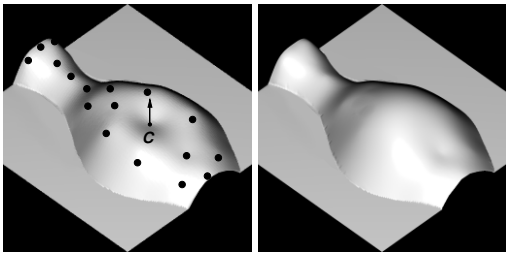
The region within the boundary, $\mathcal{B}_W$ is sampled finely on a discrete grid $\mathcal{G}$, where for each grid location $(x_g, y_g) \in \mathcal{G}$, the depth z_g and the surface normal $\mathbf{N}(x_g, y_g, z_g)$ are computed from the fitted surface. The integral in (1) is discretized in a similar fashion by considering a window of size $4pq$ around the projected pixel m and averaging out the mean square error. Note that handling the variational problem in a quantitative domain allows us to effectively handle the *finite light source* scenario without getting into complex analytical expressions. The reinforcement β thus turns out to be:

$$\beta = 1.0 - \frac{C(\mathbf{S}, \mathbf{N})}{\mid \mathcal{G} \mid} \qquad (14)$$

The algorithm is run until the convergence conditions listed in section 3 are met. This concludes the description of the basic algorithm. In the next section, we describe an interactive setting that allows us to deal with the SFS local minima in an effective manner.

5. User Interaction

After the initial user interaction/input, the algorithm proceeds to converge to a *local minimum* as noted in section 3. At this stage, the user requirements govern whether the result obtained is acceptable as is or a better result is expected. In the latter case, the user can interact with the resultant 3D mesh $\mathcal{S}$ to pull the system out of the local minimum and let the algorithm converge to another local minimum or possibly a *global minimum*. The interaction scheme basically allows the user to adjust the depth values at each of the points $\{(x, y, z) \in \mathcal{S} : (x, y) \in \mathcal{C}_W\}$ following which the μ values for the altered CALA are shifted to the new z values (keeping their σ unaltered) and the algorithm is resumed. The user can also add to the set of points $\mathcal{C}_I$ (in case the details in a particular region need to be enhanced) or delete from the existing set (to impart smoothness to a local region around the deleted location). For the newly added points, the μ values are assigned by interpolating from the current surface $\mathcal{S}$ while the σ values are specified by the user.



(a) (b)

Figure 3. Illustration of User Interaction: (a) Before Interaction and (b) surface after pulling on CALA C. Black dots indicate locations in $\mathcal{C}_W$.

Fig.3 illustrates this idea at an intermediate stage of reconstruction of the surface of a vase, where the algorithm showed no significant change in the *payoff* value β, indicating a *local minimum*. A slight trough is visible on the surface at the location marked C in Fig.3(a), indicating a deviation from the expected result. The arrow depicts the direction and magnitude of change in the depth value at the said location initiated by the user to pull the system out of the local minimum. Fig.3(b) depicts the surface after the algorithm has adjusted to the interaction. Note that this setting is the key to the claimed *incremental* framework. The algorithm can take in additional constraints and use them in conjunction with the learned history, without the need for a restart.

Note that a change δ in the mean value μ_i of a CALA C_i s.t. $\delta \in (\mu_i - \sigma_i, \mu_i + \sigma_i)$ describes a change in the playing strategy chosen by player i (CALA C_i). From the definition of *Nash Equilibrium* in (8), it is clear that the system cannot be pulled out of a local minimum, if player i is the only CALA whose μ is updated by the user at the minimum. Thus, the change initiated by the user must be large enough (outside the probability hill) to ensure that the chosen strategy does not belong to the current strategy set of C_i. The system would then be able to overcome the minimum.

6. Hierarchical Reconstruction

We propose a scheme of *hierarchical reconstruction* for complex surfaces which cannot be suitably represented as an interpolating *thin plate functional* through a set of 3D points. One such example is a face, where features like nose, lips etc., preclude dealing with the whole face as a single thin plate functional. We illustrate the procedure on the classical Mozart face in Fig.4. Instead of dealing with

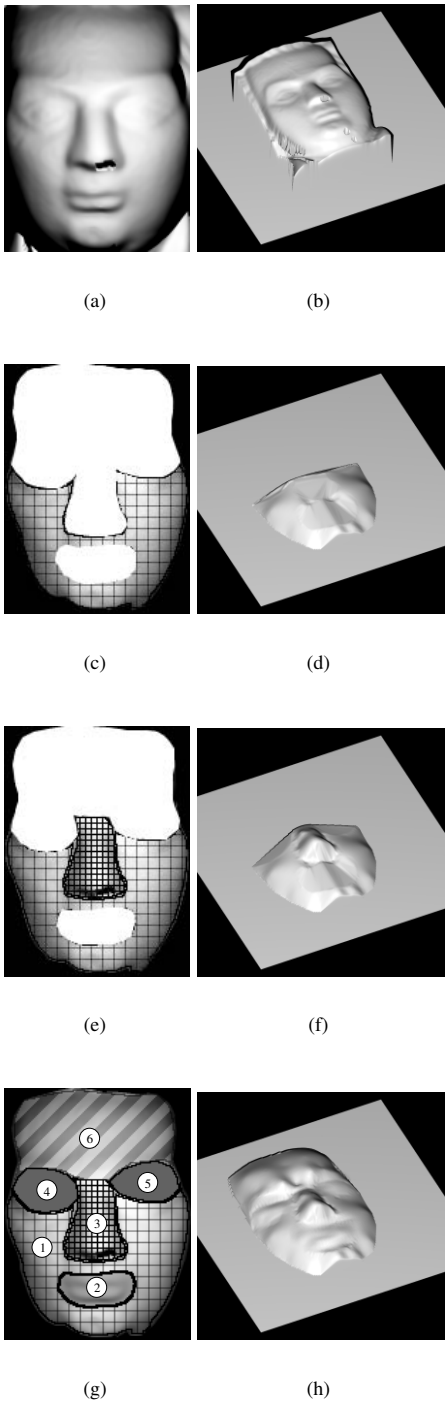


Figure 4. Hierarchical Reconstruction of Mozart's face: (a)&(b) Original image and surface, (c)&(d) level-1 reconstruction, (e)&(f) level-2 reconstruction and (g)&(h) final reconstruction.

the whole face as one instance, the user segments out each sub-region of the face and specifies the sets $\mathcal{B}_I$ and $\mathcal{C}_I$ and the mask, separately for each of the regions as shown in Fig.4(g) where the number of sub-regions is 6. The algorithm is then run sequentially on each sub-region, by including only the image pixels within the mask of that sub-region, for the payoff computation. The reconstructed surface for each sub-region acts as a seed for initializing the μ values of the boundary points of the next (adjoining) sub-region under consideration. In this manner, the method allows dealing with complex surfaces while maintaining C^0 continuity along creases. In the first stage, reconstruction of sub-region 1 shown in Fig.4(c) results in the surface in Fig.4(d). The adjoining sub-region (nose) 3 is considered next. The boundary height values for sub-region 3 are initialized from the surface obtained for sub-region 1 and the reconstruction algorithm run for this sub-region. The merged result for the two sub-regions is shown in Fig.4(f). Finally, Fig.4(h) shows the complete reconstructed surface.

7. Finer Aspects

In this section, we present some enhancements to our basic algorithm that allow for better control over the reconstructed surface and help in speeding up the algorithm.

7.1. Placement of CALA

As one can expect, the convergence rate of the CALA algorithm is directly governed by the number of locations in $\mathcal{C}_I$. It is therefore expected that these locations be marked out in a topology aware manner, as far as possible. Consider the lateral view of a *surface of revolution* as an example. After the boundary of the SOR is marked out and held fixed, populating $\mathcal{C}_I$ with more number of points on the axis of revolution and fewer around the axis serves the same purpose as having a uniform density of points all around, though the former takes much less time to converge. Also, since the surface has a degree of freedom at the marked locations only, more locations need to be marked in regions where finer detail is required.

7.2. Pyramidal Refinement

As noted before, $|\ \mathcal{C}_W\ |$ governs the convergence rate of the algorithm. We propose a *pyramidal* scheme in which the algorithm starts from a subset $\mathcal{C}_W^* \subset \mathcal{C}_W$ and converges to a crude solution surface. The set $\mathcal{C}_W^*$ being smaller, this stage of the process is fast. In the next stage, the set $\mathcal{C}_W^*$ is broadened to include a larger number of points from $\mathcal{C}_W$ and the algorithm is resumed. The initial μ values for the newly added points (CALA) is arrived at from the smooth surface $\mathcal{S}$ fitted through the points $\{(x,y,\mu(x,y)) : (x,y) \in$

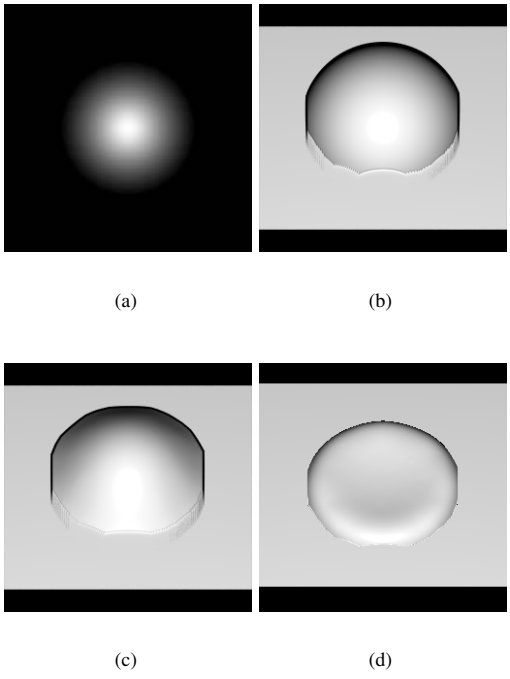


Figure 5. Sphere ($l = (0,0,3.5)$,$f = 1.0$): (a) Original image, (b) original surface, (c) reconstructed surface and (d) the error surface.

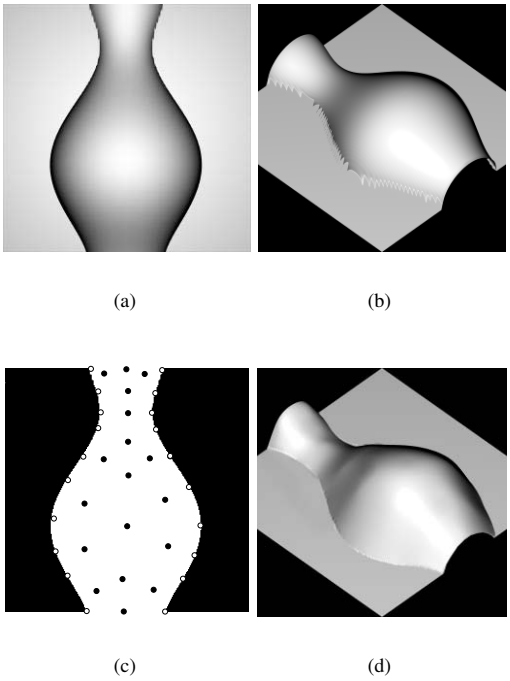


Figure 6. Vase: (a) Original image, (b) original surface, (c) points in the sets $\mathcal{B}_I$ (white circles) and $\mathcal{C}_I$ (black circles) and (d) the reconstructed surface.

$\mathcal{C}_W^*\}$ where $\mu(x,y)$ is the mean value obtained by CALA at location (x,y) at the end of the previous stage.

8. Results

The scheme proposed has been extensively tested for a number of cases. Firstly the simple case of a spherical lambertian surface as depicted in Fig.5 was considered where the algorithm managed to converge to an acceptable solution without any intervention from the user. User interaction was necessary for the case of the vase, as already indicated in section 5. The final results for the vase are presented in Fig.6.

In the context of hierarchical reconstruction (section 6), we have already illustrated the results for case of the Classical Mozart face, see Fig.4.

It is to be noted that in all these cases the fixed points on the boundary were identified from the silhouette in the image and were associated with the same average depth. This is but an approximation in the context of perspective imaging as the points on the silhouette should be associated with the occluding contour [13]. Although this approximation does affect the quality of results, but the solutions obtained are acceptable.

The estimation of errors in the reconstruction has little meaning when one allows interaction. Thus we considered the case of a spherical surface (where no user interaction was needed) and have compared the results obtained with the ground truth. The absolute error was computed on the sampled points and the error surface is presented in Fig.5(d). In this case the boundary points were associated with the actual depth obtained from ground truth. The error under norm $||\cdot||_2$ was of the order of 3%.

As can be expected such a learning based SFS scheme is slow. The complete vase took approximately an hour to reach the solution on a Pentium IV 3.06 GHz machine. However, this is not a crucial issue. Firstly the scheme described in the contribution requires user intervention only in stages. The user needs to specify the initial configurations and then can allow the algorithm to converge to a minimum. The user can then intervene if necessary to make alterations to the set of CALA and resume the algorithm. Secondly, as indicated in section 7 the number of CALA can be reduced by placing fewer CALA at judicious locations to enhance the speed. Also as indicated earlier the paralleliz-

ability of the CALA algorithm can be suitably exploited to obtain greater speed-ups.

9. Conclusion

In this contribution, we have proposed a novel scheme for the reconstruction of free-form lambertian surfaces. Our approach allows user interaction when required, to help the algorithm reach an acceptable solution. Being essentially a non-analytical framework, the method handles perspective as well as orthographic cases and also light sources placed at finite distances from the object. Features such as progressive reconstruction and pyramidal refinement make the task of modeling easier.

We are extending our method to handle different models of shading and incorporate aspects like shadows, interreflections, specularity etc. Learning the light source location is also a possible direction of improvement.

References

[1] A. Blake and C. Marinos. Shape from texture: estimation, isotropy and moments. *Artificial Intelligence*, 45(3):323–80, 1990.

[2] D.Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. *EUROGRAPHICS*, 1999.

[3] O. Faugeras and R. Keriven. Variational principles, surface evolution, pde's, level set methods and the stereo problem. *Technical Report 3021, INRIA*, November 1996.

[4] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[5] B. Horn. Shape from shading: A method for obtaining the shape of a smooth opaque object from one view. *MIT AI-TR*, 1970.

[6] B. K. P. Horn. *Robot Vision*. MIT Press, 1986.

[7] J. Koenderink. Pictorial relief. *Phil. Trans. of the Roy. Soc: Math., Phys., and Engineering Sciences*, 356(1740):1071–1086, 1998.

[8] E. Prados and O. Faugeras. Perspective shape from shading and viscosity solutions. *IEEE Proceedings of ICCV'03*, 2:826–831, October 2003.

[9] G. Santharam, P. Sastry, and M. Thathachar. Continuous action set learning automata for stochastic optimization. *Journal of the Franklin Institute*, 5(331B):607–628, 1994.

[10] P. S. Sastry, V. V. Phansalkar, and M. A. L. Thathachar. Decentralised learning of nash equilibria in multi-person stochastic games with incomplete information. *IEEE Transactions on Systems, Man and Cybernetics*, 24:769–777, May 1994.

[11] P. S. Sastry, K. Rajaraman, and S. R. Ranjan. Learning optimal conjunctive concepts through a team of stochastic automata. *IEEE Transactions on Systems, Man and Cybernetics*, (23):1175–1184, 1993.

[12] M. A. L. Thathachar and M. T. Arvind. Parallel algorithms for modules of learning automata. *IEEE Transactions on Systems, Man and Cybernetics*, Part B, 28:24–33, 1998.

[13] K.-Y. K. Wong, P. R. S. Mendonça, and R. Cipolla. Reconstruction of surfaces of revolution from single uncalibrated views. *BMVC*, 2002.

[14] L. Zhang, G. Dugas-Phocion, J.-S. Samson, and S. M. Seitz. Single view modeling of free-form scenes. *Proc. Computer Vision and Pattern Recognition*, 2001.

[15] R. Zhang, P.-S. Tsai, J. Cryer, and M. Shah. Shape from shading: A survey. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 21(8):690–706, August 1999.