Training Many-Parameter Shape-from-Shading Models Using a Surface Database

Nazar Khan
University of Central Florida
nazar@cs.ucf.edu

Lam Tran
University of California, San Diego
lat003@ucsd.edu

Marshall Tappen
University of Central Florida
mtappen@eecs.ucf.edu

Abstract

Shape-from-shading (SFS) methods tend to rely on models with few parameters because these parameters need to be hand-tuned. This limits the number of different cues that the SFS problem can exploit. In this paper, we show how machine learning can be applied to an SFS model with a large number of parameters. Our system learns a set of weighting parameters that use the intensity of each pixel in the image to gauge the importance of that pixel in the shape reconstruction process. We show empirically that this leads to a significant increase in the accuracy of the recovered surfaces. Our learning approach is novel in that the parameters are optimized with respect to actual surface output by the system.

In the first, offline phase, a hemisphere is rendered using a known illumination direction. The isophotes in the resulting reflectance map are then modelled using Gaussian mixtures to obtain a parametric representation of the isophotes. This Gaussian parameterization is then used in the second phase to learn intensity-based weights using a database of 3D shapes. The weights can also be optimized for a particular input image.

1. Introduction

Shape-from-shading (SFS) attempts to reconstruct the shape of a three-dimensional object from its shading in a two-dimensional image. This paper presents a parametric example-based approach for SFS that incorporates machine learning to improve reconstruction accuracy. Learning from training data has had considerable impact on areas such as recognition [2] and low-level vision [5, 17]. Despite successful incorporation of learning into so much of computer vision, there has been little use of training data to directly optimize the reconstructions from SFS systems.

A wide variety of solutions have been proposed for the classic SFS problem. Survey studies in [4, 24] have broadly grouped solutions into three classes of methods based on: partial differential equations [16], optimization [7, 23] and image irradiance equation approximation [20]. Despite their differences, SFS solutions are similar in that they are based on underlying mathematical formulations with a handful of parameters. In optimization-based methods, these parameters are weighting parameters for penalty terms that impose constraints such as smoothness and integrability [4, 24].

Shape-from-shading methods tend to rely on models with few parameters because these parameters need to be hand-tuned, thus limiting the number of different parameter-value combinations that can be evaluated. Using brute-force grid search methods suffer a similar limitation as the number of different value combinations that must be evaluated grows exponentially with the number of parameters.

In this paper, we show how machine learning can be applied to an SFS model with a large number of parameters. We show in Section 5.1 how adding a larger number of parameters that assign intensity-based weights to input pixels leads to significant gains in the accuracy of the system. While the proposed approach has limitations, which are discussed in Section 7, this learning-based approach has the potential to enable significant innovations on SFS problems. The ability to search over large parameter spaces in an automated fashion makes it possible to train complex models that use many different types of features. The benefits of this capability can be seen in the recent progress on object recognition, where learning is integral to state-of-the-art methods [2, 9].

2. Related Work

In previous work on learning and shape-from-shading, the term learning has been used in two ways. In works like [3, 22, 10], neural network learning algorithms are adapted to perform the optimization necessary to produce a surface estimate. In this context, the system learns a set of parameters that are tuned to reconstruct a surface from a single example. It should be noted that no ground-truth data is used in these systems as the learning can be thought of as an alternative surface optimization technique.

In our work, we use the term learning as in the context of supervised learning. Using a database of examples, our system learns the model parameters that lead to the best recon-

2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops
978-1-4244-4441-0/09/$25.00 ©2009 IEEE

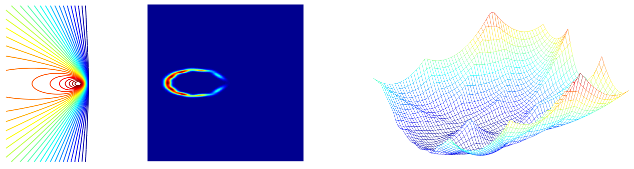

(a) (b) (c)

Figure 1: Example of isophotes and constraints. (a) Some of the isophotes in a Lambertian reflectance map. (b) A Gaussian Mixture Model consisting of 7 mixtures fit to one of the isophotes. (c) The negative log-likelihood of (b).

struction possible. In contrast to the works described above, these parameters are used for all images. This style of learning has been proposed in [13] where the system learns local estimators that predict orientation using local data. Orientation estimates are also used in [12] to recover an estimate of the final surface.

Our approach is novel in that it goes beyond learning of local estimators. Instead, the output of the entire system is holistically trained. We use the term holistic because every parameter is optimized with respect to the final estimate of the surface depth, in contrast to [12] and [1] where the training is used to optimize intermediate and/or local estimates. As will be discussed in Section 5, this is made possible by the novel application of the Variational Mode Learning framework [19].

3. Basic Model

We solve the SFS problem with known illumination direction, constant albedo and an orthogonal camera. Following the broad groupings from the comparison papers by Zhang et al. and Durou et al. [4, 24], our proposed SFS method is an optimization based approach. Accordingly, we define an energy function $E(\mathbf{z}, \theta)$, with parameters θ, over shapes $\mathbf{z}$. This energy function $E(\mathbf{z}, \theta)$ will be minimized to find the estimate, $\mathbf{z}^*$, of an object's shape.

Our basic approach is to formulate the energy function $E(\mathbf{z}, \theta)$ in a manner that makes it possible to use the Variational Mode Learning (VML) algorithm [19] to learn the parameters θ. VML was introduced for learning the parameters of Markov Random Fields. Using a set of ground-truth shapes, we employ VML to find the parameters θ that minimize the difference between the ground-truth shapes and the estimates returned by the SFS system.

To find the parameters θ, we define a loss function $L(\mathbf{z}^*, \mathbf{t})$ that measures the difference between the estimate, $\mathbf{z}^*$, returned by the SFS system, and the ground-truth shape $\mathbf{t}$. The VML approach is used to calculate the derivative of this loss function with respect to θ and find the optimal parameters θ using gradient descent optimization[1].

To accommodate the VML algorithm, the data term of $E(\cdot)$ is formulated as the negative log-likelihood of a mixture of Gaussian models. Formulating the data-term in this fashion allows us to minimize $E(\mathbf{z}, \theta)$ by minimizing a series of quadratic upper-bounds on $E(\mathbf{z}, \theta)$ to find $\mathbf{z}^*$. This is a series of differentiable operations. Thus the result, $\mathbf{z}^*$, of the minimization can be differentiated with respect to θ and used to compute the gradient of $L(\cdot)$ with respect to θ.

[1]The overall training criterion is non-convex with respect to θ, so a set of parameter values could be a local minimum of the training criterion

4. Implementing the Data Term

In shape-from-shading, the intensity of a pixel in the image constrains the surface normal at the corresponding point to lie along a curve in the reflectance map of possible surface orientations. The isophotes of the reflectance map define the possible orientations that a surface point with a specific intensity can take. We construct the data term by fitting Gaussian Mixture models to the isophotes corresponding to different image intensity values.

Below, Section 4.1 describes how the mixture models are created. Section 4.2 describes how the data term is constructed from these Gaussian mixture models and introduces the formal notation that will be used throughout the rest of the paper.

4.1. Fitting the Mixture Models

To begin, we discretize the range of possible surface intensities into B intervals, or bins. For each intensity bin b, we sample a number of locations in the reflectance map where the intensity falls in bin b. Each of these locations corresponds to a valid orientation, which we express using the derivatives of the surface height. Following convention, we refer to these derivatives as p and q. After generating these locations, the Expectation Maximization (EM) algorithm is used to fit a Gaussian Mixture Model to this set of surface normals. While this process must be repeated for every illumination used to render the input images, this step must only be performed once and can thus be precomputed for many different illuminations.

Figure 1 illustrates the process of generating the mixture models for the data term. Figure 1(a) shows some of the isophotes in a Lambertian reflectance map. Figure 1(b) shows a Gaussian Mixture Model fit to the surface orientations consistent with a specific range of surface intensities. Here, a mixture model with seven components is shown. Figure 1(c) shows the negative log-likelihood of (b).

4.2. Constructing the Data Term Energy Function

We start with some notation. Let l_j be the intensity bin corresponding to the intensity at pixel j, $\mathcal{C}(l_j)$ be the set of Gaussian mixture components corresponding to bin l_j. As described above, a mixture model is fit for each intensity bin. In each mixture $\mathcal{C}(l_j)$, we define i_c to be the cth component from $\mathcal{C}(l_i)$, $\boldsymbol{\mu}_{i_c}$ be the 2×1 mean vector of component i_c, $\boldsymbol{\Sigma}_{i_c}^{-1}$ be the 2×2 precision matrix of component i_c, K_{i_c} be the various constants, including the mixing coefficient and the normalization term of component i_c and $\mathbf{g}_i = [p_i, q_i]^T = [\frac{\partial z_i}{\partial x}, \frac{\partial z_i}{\partial y}]^T$ be the gradient vector of the shape $\mathbf{z}$ at pixel i. $\mathbf{z}$ is a column-vector representation of the shape. The quadratic component in the exponent of each mixture component can then be expressed as

$$Q_{i_c} = -\frac{1}{2}\left(\mathbf{g}_i - \boldsymbol{\mu}_{i_c}\right)^T \boldsymbol{\Sigma}_{i_c}^{-1}\left(\mathbf{g}_i - \boldsymbol{\mu}_{i_c}\right). \tag{1}$$

We can write the negative log-probability of any particular values of p_i and q_i as

$$-\log P(p_i, q_i) = -\log \sum_{c=1}^{N_c} \exp(Q_{i_c} + \log K_{i_c}), \tag{2}$$

where N_c is the number of Gaussian mixture components. For the remainder of this paper, we will describe our approach using an energy-minimization criterion. Thus we can think of Equation (2) as defining an energy function over possible orientations at point i of shape $\mathbf{z}$. This allows us to write our energy functional as

$$\begin{aligned} E(\mathbf{z}) = & \underbrace{\sum_{i=1}^{Np} -\log \sum_{c=1}^{N_c} \exp(Q_{i_c} + \log K_{i_c})}_{\text{data term } E_d} \\ & \underbrace{+ \lambda_s \sum_{i=1}^{Np} |\nabla p_i|^2 + |\nabla q_i|^2 + \alpha z_i^2,}_{\text{smoothness term } E_s} \end{aligned} \tag{3}$$

where N_p is the total number of pixels, and λ_s is the smoothness parameter. The term αz_i^2 is added for numerical stability with the weight α set to a very low value of 1×10^{-6} to avoid flat surfaces. Since minimization of the quadratic smoothness term is straight-forward, we focus in the following on the minimization of the data term E_d.

4.3. Quadratic Upper-bounds on the Energy

Equation (2) is non-convex, with many local minima. We can upper-bound Equation (2), with a tight quadratic upper-bound using Jensen's inequality.

The upper-bound will be computed at a particular value of p_i and q_i, denoted by p'_i and q'_i. Accordingly, the quadratic exponents are denoted as Q_{i_c} and $Q_{i'_c}$. Using Jensen's inequality, a quadratic upper-bound on Equation (2) is then obtained as

$$-\log P(p_i, q_i) \leq \sum_{c=1}^{N_c} \frac{\exp(Q_{i'_c} + \log K_{i'_c})}{\sum_{j=1}^{N_c} \exp(Q_{i'_j} + \log K_{i'_j})} Q_{i_c} + T, \tag{4}$$

where T denotes several constant terms that have been left out due to space considerations. We can ignore these constant terms because our final goal is to differentiate and solve this upper-bound. Thus the constant terms will not affect the final answer.

4.3.1 Expressing the Data Term at Every Pixel

The quadratic upper-bound in Equation 4 refers to the orientation at just a single pixel. This can be extended to a matrix formulation for every pixel in the image. To do so, we first define the precision matrix for a mixture component's quadratic exponent, Q_{i_c}, as

$$\boldsymbol{\Sigma}_{i_c}^{-1} \triangleq \begin{bmatrix} a_{i_c} & b_{i_c} \\ b_{i_c} & d_{i_c} \end{bmatrix}, \tag{5}$$

which allows us to express Equation (1) as a sum of terms,

$$Q_{i_c} = -\frac{1}{2}\left(a_{i_c}\hat{p}_{i_c}^2 + 2b_{i_c}\hat{p}_{i_c}\hat{q}_{i_c} + c_{i_c}\hat{q}_{i_c}^2\right) \tag{6}$$

where $\hat{p}_{i_c} = p_i - \mu_{i_{c_p}}$ and $\hat{q}_{i_c} = q_i - \mu_{i_{c_q}}$.

Since we actually want to recover a height map, $\mathbf{z}$, we can write

$$\mathbf{p} = D_X \mathbf{z}, \tag{7}$$

$$\mathbf{q} = D_Y \mathbf{z}, \tag{8}$$

where D_X and D_Y are discrete derivative matrices that allow vectors $\mathbf{p}$ and $\mathbf{q}$ to hold the values of p and q at every pixel in the shape vector $\mathbf{z}$. This substitution makes it possible to directly solve for $\mathbf{z}$. Thus, our method directly computes the height-map without requiring a second integration step.

We can then compute the vectors

$$\hat{\mathbf{p}}_c = \mathbf{p} - \boldsymbol{\mu}_{c_p}, \tag{9}$$

$$\hat{\mathbf{q}}_c = \mathbf{q} - \boldsymbol{\mu}_{c_q}, \tag{10}$$

where $\boldsymbol{\mu}_{c_p}$ (resp. $\boldsymbol{\mu}_{c_q}$) is the vector of the mean horizontal (resp. vertical) gradient of the cth component of all pixels. This allows us to represent the vector of quadratic exponents at each pixel as

$$\mathbf{Q}_c = -\frac{1}{2}\left[\hat{\mathbf{p}}_c^T A_c W_c \hat{\mathbf{p}}_c + 2\hat{\mathbf{p}}_c^T B_c W_c \hat{\mathbf{q}}_c + \hat{\mathbf{q}}_c^T D_c W_c \hat{\mathbf{q}}_c\right] \tag{11}$$

where A_c is a diagonal matrix such that the ith entry along the diagonal, $[A_c]_{i,i}$ is equal to the a_{i_c} from $\boldsymbol{\Sigma}_{i_c}^{-1}$ in Equation 5. The matrices B_c and D_c are similarly defined using b_{i_c} and d_{i_c}. The matrix W_c is also a diagonal matrix, with the ith entry along the diagonal defined as

$$[W_c]_{i,i} = \frac{\exp\left(Q_{i'_c} + \log K_{i'_c}\right)}{\sum_{j=1}^{N_c} \exp\left(Q_{i'_j} + \log K_{i'_j}\right)} \tag{12}$$

So, an upper-bound on the data term E_d for every pixel, which we will refer to as $\hat{E}_d(\mathbf{p}, \mathbf{q})$ can be written in matrix form as

$$\hat{E}_d(\mathbf{p}, \mathbf{q}; \mathbf{p}', \mathbf{q}') = \sum_{c=1}^{N_c} -\mathbf{Q}_c + \mathbf{T} \tag{13}$$

where the vector $\mathbf{T}$ consists of constant terms that we need not worry about for minimization purposes.

Essentially, every row in these matrices corresponds to the quadratic upper-bound of a mixture component at a particular pixel. It can be shown that this upper-bound is tight, i.e. $\hat{E}_d(\mathbf{p}', \mathbf{q}'; \mathbf{p}', \mathbf{q}') = E_d(\mathbf{p}', \mathbf{q}')$. Thus, the vectors $\mathbf{p}'$ and $\mathbf{q}'$ can be thought of as the point where the upper-bound is being computed and will touch the actual function. The matrix W_c can be thought of as a weighting term that weights the different quadratic components to produce a convex upper-bound.

4.4. Recovering Height-Maps with Coordinate Descent

Our primary motivation for introducing this upper-bound is to use it to minimize the energy function. Because the upper-bound is tight, we optimize the heightmap $\mathbf{z}$ such that the energy never increases. This permits us to train the system parameters to directly optimize the result of the minimization. Given an estimate of $\mathbf{z}$ at iteration t, denoted $\mathbf{z}^t$, the next estimate is found using these steps:

1. For all mixture components, $c = 1 \dots N_c$, compute $\mathbf{p}' = D_X \mathbf{z}^t$, $\mathbf{q}' = D_Y \mathbf{z}^t$ and use them to compute W_c. Effectively, the upper-bound, $\hat{E}_d(\mathbf{z}^{t+1}; \mathbf{z}^t)$ will be recomputed at $\mathbf{z}^t$.

2. Obtain new estimate $\mathbf{z}^{t+1}$ by minimizing Equation (13).

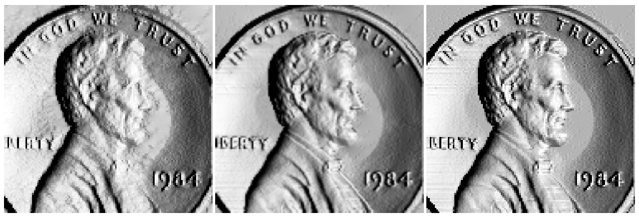
We refer to this approach as coordinate descent because computing the $Q(\cdot)$ terms in Step 1 can be viewed as minimizing variational parameters [11]. Since $\hat{E}_d(\mathbf{z}^{t+1}; \mathbf{z}^t)$ is quadratic, step 2 can be solved using standard least-squares techniques and $\mathbf{z}^{t+1}$ can be computed by solving a linear system.

This optimization is similar to the Expectation-Maximization algorithm that is popular for finding parameters for Gaussian Mixture models. At a high level, this method is similar to that proposed in [8], in that the system is iteratively choosing between different quadratic "exemplar" models of the gradient at each point.

4.5. Evaluating the Shape-Estimation System

Before describing how we can incorporate parameter learning into this approach, we will first evaluate this basic system against recently proposed approaches. Our intent is to show that even without 3D ground-truth based training, this approach can produce competitive results. Figure 2 shows the rendered result from our system on the penny surface from [24], compared with rendered height map from Potetz's method [15]. Our result[2] is qualitatively close to [15] but is obtained much faster.

[2]Obtained using the mean squared error between the rendered reconstruction and the input image as the loss function (Section 5)



(a) Our result. MSE=463. Time=8.6m (b) Potetz's result. MSE=108. Time=24h (c) Ground-truth.

Figure 2: A baseline comparison of (a) our method with (b) Potetz's method [15] (Image reproduced from [15]).

Table 1 provides a quantitative comparison of our system with [6] and [23] for the reconstruction of the Mozart surface. Each column shows the percentage of normals that are within the given angular difference from the ground-truth. Even without learning the weighting parameters, the system presented so far compares favorably against [6] and [23] while for the more relaxed angular errors, the trained system is quantitatively superior. In the next section, we describe how weighting parameters are incorporated and learned.

5. Model Improvement Through Learning

While our basic system performs well, the most powerful aspect of our approach is that it is possible to learn the parameters of the system. In section 5.1, we introduce a new set of weighting parameters to the energy function in Equation 3. Section 5.2 will then discuss how to learn these weighting parameters using Variational Mode Learning.

5.1. Weighing Intensity Intervals

As described in Section 4.2, we construct our model by fitting a Gaussian Mixture to the possible surface orientations, parameterized by derivative values. Using the learning framework described below, we modified the energy function in Equation 3 to allow us to weight the different intensity ranges differently. Our reasoning is that some intensity ranges provide more reliable information than others. For instance, in a Lambertian reflectance map, the orientation of points with the brightest intensity can be unambiguously identified. In this case, the isophote in the reflectance map is a single point. Because the surface normal is known, it would seem logical that the energy functions constraining the normals at these points would receive higher weight. This higher weight would reflect that there is less ambiguity in the estimate of surface normal [3]

We implement this weighting by modifying the data term E_d in Equation 3 to include weighting terms:

$$E_d(\mathbf{z}) = \sum_{i=1}^{N_p} -\exp(w_i) \log \sum_{c=1}^{N_c} \exp\left(Q_{i_c} + \log K_{i_c}\right) \tag{14}$$

[3]While this scheme is intuitively motivated, in the following sections we show strong experimental evidence demonstrating the efficacy of this weighting scheme.

Angular difference	1°	2°	3°	4°	5°	10°	15°	20°	25°
Haines & Wilson [6]	0.2	0.8	2.1	4.5	7.9	21.9	33.3	43.1	50.4
Worthington & Hancock [23]	2.4	5.4	8.0	10.4	13.4	25.0	33.4	40.5	46.8
Presented system (untrained)	0.6	2.0	4.0	6.6	9.4	23.2	36.8	48.3	59.0
Presented system (trained)	0.6	2.0	4.2	7.0	10.3	26.2	41.3	54.5	65.9

Table 1: Percentages of normals of the Mozart reconstruction that are within the given angular difference (column-wise) from the ground-truth normals. Illumination direction given by (-1, 0, 1). Our untrained system is comparable to [6] and [23] while the trained one is quantitatively superior for more relaxed angular errors.

where w_i is the weight associated with the intensity-bin $l(i)$ corresponding to the intensity at pixel i. We include the exponential to insure that the final weight is positive. This enables us to do an unconstrained minimization when learning the weighting parameters. When fitting the upper-bounds (Section 4.3) during optimization, these weights will be multiplied with the quadratic upper-bound in a fashion similar to Equation 14.

5.2. Variational Mode Learning

Having defined weighting parameters for different intensity ranges, we now describe how these parameters can be optimized. The first step is to define a loss function that evaluates the quality of the result returned using any particular set of parameters. We use the loss on surface orientation which is defined as:

$$L(\mathbf{z}, \mathbf{t}) \triangleq \sum_{i=1}^{N_p} 1 - \mathbf{n}_i^{\mathbf{z}} \cdot \mathbf{n}_i^{\mathbf{t}} \qquad (15)$$

where $\mathbf{t}$ is the ground-truth and $\mathbf{n}_i^{\mathbf{z}}$ is the normalized normal vector to the surface $\mathbf{z}$ at pixel i. We can also use the mean squared error between the rendered reconstruction and the input image as the loss function. This allows the algorithm to work in an optimization framework without ground-truth shapes like [3, 22, 10] rather than in a learning framework.

Once the loss function has been defined, we can use the Variational Mode Learning (VML) technique [19] to find the parameters that minimize the loss for the heightmap estimated after running a fixed number of coordinate descent iterations described in Section 4.4.

Formally, if $\mathbf{z}^*$ is the output of N_I coordinate descent steps, the goal is to find the weighting parameter vector $\mathbf{w} = (w_1, w_2, \ldots)$ which minimize $L(\mathbf{z}^*, \mathbf{t})$. This can be accomplished using straightforward gradient-based optimization – if it is possible to compute the gradient of $L(\mathbf{z}^*, \mathbf{t})$ with respect to $\mathbf{w}$.

Since $\mathbf{z}^*$ is the result of a set of differentiable coordinate descent steps, we can compute the gradient of $L(\mathbf{z}^*, \mathbf{t})$, $\frac{\partial L}{\partial \mathbf{w}}$, using the chain rule in a style very similar to back-propagation. This is described next.

5.3. Basic Learning Algorithm

If $\mathbf{z}^*$, which we will also denote as $\mathbf{z}^{N_I}$, is the result of N_I coordinate descent steps, we use a set of recursive steps to compute $\frac{\partial L}{\partial \mathbf{w}}$. The underlying idea behind VML is that by applying the chain rule, the derivative of $L(\cdot)$ with respect to some parameter w_i can be computed as

$$\frac{\partial L}{\partial w_i} = \sum_{n=1}^{N_I} \frac{\partial L}{\partial \mathbf{z}^n}^T \frac{\partial \mathbf{z}^n}{\partial w_i} \qquad (16)$$

where the intermediate values of $\mathbf{z}$ after each iteration are labeled $\mathbf{z}^1, \mathbf{z}^2, \ldots, \mathbf{z}^{N_I}$.

The partial derivatives in this summation can be computed efficiently in a recursive fashion, using the following steps:

1. Initialize $\frac{\partial L}{\partial \mathbf{w}}$ to all zeros.
2. Compute $\frac{\partial L}{\partial \mathbf{z}^{N_I}}$.
3. For $n = N_I \ldots 0$:
   (a) $\frac{\partial L}{\partial \mathbf{w}} \leftarrow \frac{\partial L}{\partial \mathbf{w}} + \frac{\partial L}{\partial \mathbf{z}^n} \frac{\partial \mathbf{z}^n(\mathbf{z}_F^{n-1}, \mathbf{w})}{\partial \mathbf{w}}$ Notice that we have appended an "F", for fixed, to $\frac{\partial \mathbf{z}^n(\mathbf{z}_F^{n-1}, \mathbf{w})}{\partial \mathbf{w}}$ and added an argument. This is to indicate that in these computations $\mathbf{z}^{n-1}$ is treated as a constant on which $\mathbf{z}^n$ depends.
   (b) If $n > 0$, $\frac{\partial L}{\partial \mathbf{z}^{n-1}} \leftarrow \frac{\partial L}{\partial \mathbf{z}^n} \frac{\partial \mathbf{z}^n}{\partial \mathbf{z}^{n-1}}$. This final matrix, $\frac{\partial \mathbf{z}^n}{\partial \mathbf{z}^{n-1}}$ is the Jacobian matrix relating $\mathbf{z}^n$ and $\mathbf{z}^{n-1}$.

The equations for computing the various matrices are quite long. Due to space and formatting considerations, we refer the reader to [19].

6. Experiments and Results

In the following, we have not used any boundary constraints for our reconstructions.

6.1. Synthetic Surfaces

Since training sets for the shape from shading problem are not readily available, we generated smooth synthetic surfaces, 64 for training and 128 for testing. For the first experiment, we fix the illumination vector at (-1, -1, 1) and learn the weighting parameters using the training set. To achieve a good balance between performance and training time we use 5 coordinate descent iterations for heightmap estimation. We present the training samples in random order and repeat for 30 passes over the training set.

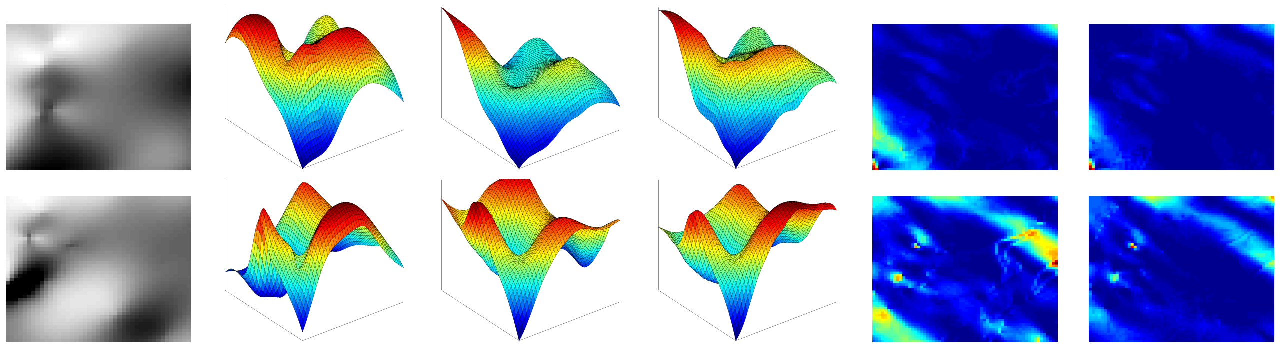

Input image
Ground-truth
Reconstruction without learning
Reconstruction with learning
Angular error without learning
Angular error with learning

Figure 3: Comparison of reconstruction with and without learning. Reconstruction with learning is perceptually closer to the ground-truth and the per pixel angular error is also lower as exhibited by the less bright pixels in the last column. Compared to the loss without learning, loss due to learning decreased by 61% and 65% for the two test surfaces shown. Total loss for the 128 test surfaces decreased by 28.7%.

λ_s	1	$\hat{\lambda}_s$
Decrease in training loss	54.0%	34.5%
Decrease in testing loss	50.1%	28.7%

Table 2: Percentage decrease in loss due to training for the cases when smoothness parameter λ_s is fixed at 1 and at its optimal value $\hat{\lambda}_s$ for the training set. It can be seen that parameter learning provides a benefit on top of using the best smoothness parameter value.

Compared to adjusting the smoothness parameter λ_s, learning bin-weights gives us many more parameters that help guide the optimization towards better numerical solutions. In order to show that learning the bin-weights has an advantage on top of adjusting the smoothness parameter λ_s, we sequentially searched for the optimal value $\hat{\lambda}_s$ that minimized the loss of the untrained system on our training set of 64 synthetic surfaces. We then trained the system using $\hat{\lambda}_s$. Table 2 shows that training loss went down by a further 34.5% due to parameter learning while testing loss on 128 test surfaces decreased by 28.7%. For $\lambda_s = 1$, training loss went down by 54.1% due to learning and testing loss went down by 50.1%.

Interestingly, we also obtained a decrease of 28.7% in testing loss for the trained $\lambda_s = 1$ system compared to the untrained $\hat{\lambda}_s$ system. This is equal to the decrease for the system trained using $\hat{\lambda}_s$ which exhibits the robustness of our learning process to the value of λ_s.

Figure 3 provides a perceptual comparison of shape reconstruction with and without learning for two of the test surfaces. In addition to being perceptually more similar to the ground-truth, the reconstructions obtained after learning have respectively, 61% and 65% lower loss than the reconstructions without learning.

6.2. Real-world Surfaces

An accurate analysis of the performance of the system on real-world surfaces can only be done when sufficient real-world training samples are available. We used a database of 6 laser scanned faces[4], renderings of which are shown in the first column of Figure 4. We trained on 5 faces with the remaining face used for testing. Training was carried out using illumination direction $(-1, -1, 1)$, 1 iteration of the gradient descent step in Section 5.3 and in each iteration, 5 coordinate descent iterations to estimate the heightmap that minimizes energy for the current parameter estimates. We presented the training samples in random order and repeated for 30 passes over the training set. Alternating the test face gives 6 different training runs. Table 3 shows the percentage decrease in loss due to training for the 6 test faces.

Face	1	2	3	4	5	6
Decrease	23.3	29.8	38.6	36.5	27.6	11.5

Table 3: Percentage decrease in loss due to training when face i is reconstructed after training on the remaining 5 faces.

The middle column of Figure 4 shows the angular errors, Equation 15, at each pixel for the reconstruction of each face without training. The last column shows reconstructions with training. The brighter areas indicate larger angle between normals of the reconstructed shape and the ground-truth. It can be seen that training results in a reconstruction with normals closer to the ground-truth. Figure 5 shows reconstructions of face 2 from Figure 4. Apart from the reduction in numerical loss, visual improvement due to learning can be noticed on the forehead and around the left jaw.

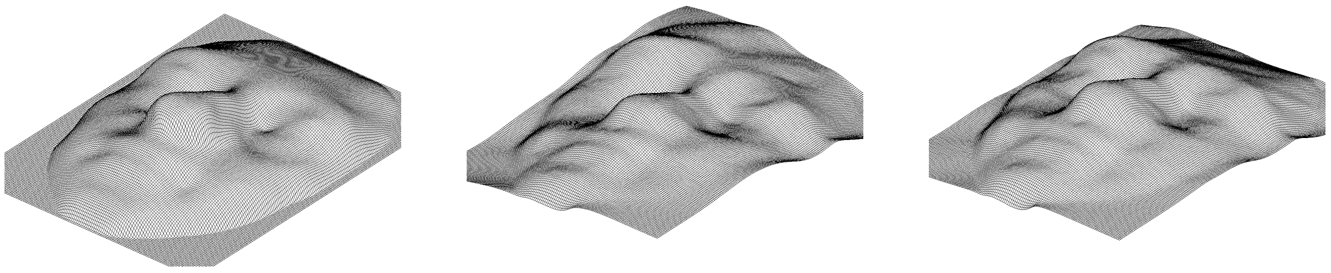
7. System Limitations

In addition to the benefits of the proposed approach, this system has limitations which are discussed in the following two sections.

7.1. Qualitative Versus Quantitative Improvement

In the previous section, we showed how training leads to significant quantitative improvement in the loss. However,

[4]We do not intend to present a facial SFS algorithm like [18]. Our choice of faces as real-world surfaces was motivated by their availability.



(a) Ground-truth. (b) Reconstruction without learning. Loss=690.54 (c) Reconstruction with learning. Loss=484.74

Figure 5: Reconstructions of face 2 from Figure 4. Apart from the reduction in numerical loss, visual improvement due to learning can be noticed on the forehead and around the left jaw.

quantitative improvement does not necessarily predict qualitatively equal improvement. This is an issue in other areas, such as image processing where the commonly used PSNR metric does not predict human perception of image quality. So researchers have devised alternate metrics, such as the recently proposed SSIM index [21].

Similarly, for SFS, a shape estimate that is significantly better, when measured quantitatively, may not be as superior qualitatively. This calls for more research on perceptually accurate 3D surface quality metrics.

7.2. Limitations from Optimization Strategy

Minimization of energy functions for SFS problems is difficult because the energy functions contain many local minima. In fact, in the model we have presented, the data term at each pixel will have multiple minima. The presence of many local minima makes the variational optimization sensitive to the initial point in the optimization. To overcome this, the system uses a consistent input as the initial point for the optimization. For lighting at relatively oblique angles, we have found the input image itself to be a suitable initial point for the optimization.

The importance of good initialization for the optimization becomes more important as the light source approaches the $[0, 0, 1]$ vector. We have found that system performance degrades significantly when the light is vertically oriented. As can be seen in Figure 1, as the light approaches the horizontal plane, isophotes begin to become straight lines in the $p - q$ map. This property was key to Pentland's work on linear shape-from-shading [14]. Straight isophotes are ideal for the variational optimization underlying the method presented here because the quadratic upper-bounds will tend to lie along the isophote, regardless of the orientation that the upper-bound is fit at.

On the other hand, if the lighting is vertical, then all of the isophotes form circles. Circular isophotes are particularly problematic for the variational optimization. Consider a system that is initialized to a flat image. In the initial image, all of the surface orientations lie at the origin of the $p - q$ map. Therefore, when the upper-bounds are fit, the quadratic functions will all be centered at the origin of the $p - q$ map. As a result, the orientations in the surface that results from minimizing these upper-bound functions will tend to stay near the origin, rather than moving to the circle that the isophote lies on. The result is a reconstruction that is overly biased towards flat surfaces pointing straight up. This leads to poorer performance on images lit with vertical illumination orientations than with more oblique orientations.

We have found that starting with a random surface that differs significantly from a flat surface somewhat alleviates this issue, though more work is needed on structuring the optimization to avoid this problem and finding better constraints. One solution is simultaneous learning of the whole system, i.e. learning the intensity-based weights alongside the Gaussian mixtures' parameters.

8. Conclusion

We have shown that given sufficient training data, the SFS problem can benefit from recent advances in Markov Random Field parameter learning techniques. SFS using parameter learning is a novel approach and it has the potential to enable significant innovations on SFS problems because of its ability to search over large parameter spaces in an automated fashion. Our method performs remarkably well for synthetic surfaces since there is no shortage of training data in that domain. Results on real-world surfaces indicate that given sufficient training data, the accuracy of SFS systems on real-world surfaces can be significantly improved.

Machine learning has been used to supplement rather than replace physically-based constraints on the SFS functional. Learning leads to significant quantitative improvement in the loss but this does not necessarily predict qualitatively equal improvement. This calls for more research on perceptually accurate 3D surface quality metrics.

References

[1] J. Ben-Arie and D. Nandy. A neural network approach for reconstructing surface shape from shading. volume 2, pages 972–976 vol.2, Oct 1998.

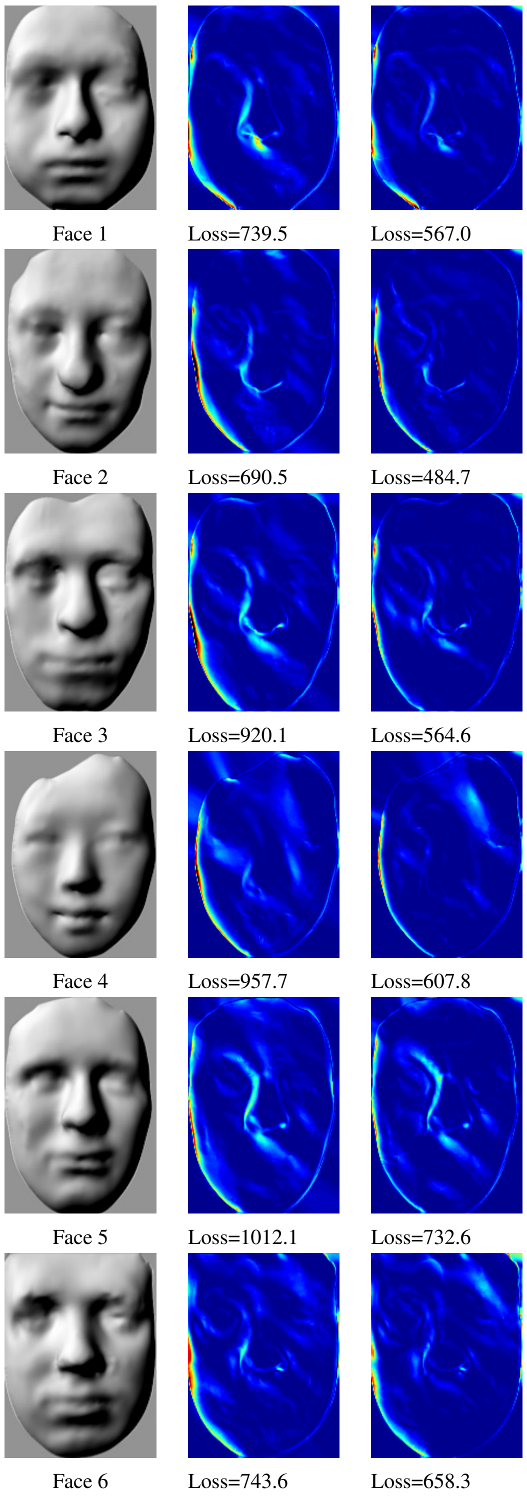


Figure 4: Database of faces. 1st column: The 6 laser-scanned faces rendered from illumination direction $(-1, -1, 1)$. 2nd column: Angular errors of reconstructions without learning. 3rd column: Angular errors of reconstructions with learning. Brighter areas signify higher angular error between reconstruction and ground-truth.

[2] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *ECCV (1)*. Springer, 2008.

[3] S.-Y. Cho and T. Chow. Neural computation approach for developing a 3d shape reconstruction model. *Neural Networks, IEEE Transactions on*, 12(5):1204–1214, Sep 2001.

[4] J.-D. Durou, M. Falcone, and M. Sagona. Numerical methods for shape-from-shading: A new survey with benchmarks. *Comput. Vis. Image Underst.*, 109(1):22–43, January 2008.

[5] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *Int. J. Comput. Vision*, 40(1):25–47, October 2000.

[6] T. Haines and R. Wilson. Belief propagation with directional statistics for solving the shape-from-shading problem. In *Proc. ECCV*, 2008.

[7] B. Horn and M. Brooks. The variational approach to shape from shading. *Computer Vision, Graphics, and Image Processing*, 33(2):174–208, February 1986.

[8] X. Huang, J. Gao, L. Wang, and R. Yang. Examplar based shape from shading. In *3DIM '07: Proceedings of the Sixth International Conference on 3-D Digital Imaging and Modeling*, August 2007.

[9] P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[10] T. Jiang, B. Liu, Y. Lu, and D. J. Evans. A neural network approach to shape from shading. *International Journal of Computer Math*, 80:433–439, 2003.

[11] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. In M. I. Jordan, editor, *Learning in Graphical Models*. MIT Press, Cambridge, 1999.

[12] D. C. Knill and D. Kersten. Learning a near-optimal estimator for surface shape from shading. *Comput. Vision Graph. Image Process.*, 50(1):75–100, 1990.

[13] S. Lehky and T. Sejnowski. Network model of sfs: Neural function arises from both receptive and projective fields. *Nature*, 333:452–454, 1988.

[14] A. P. Pentland. Linear shape from shading. *Int. J. Comput. Vision*, 4(2):153–162, 1990.

[15] B. Potetz. Efficient belief propagation for vision using linear constraint nodes. In *Proc. CVPR*, June 2007.

[16] E. Prados, F. Camilli, and O. Faugeras. A unifying and rigorous shape from shading method adapted to realistic data and applications. *Journal of Mathematical Imaging and Vision*, 25(3):307–328, October 2006.

[17] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *In CVPR*, pages 860–867, 2005.

[18] W. A. Smith and E. R. Hancock. Facial shape-from-shading and recognition using principal geodesic analysis and robust statistics. *Int. J. Comput. Vision*, 76(1):71–91, 2008.

[19] M. Tappen. Utilizing variational optimization to learn markov random fields. In *Proc. CVPR*, pages 1–8, 2007.

[20] P. S. Tsai and M. Shah. Shape from shading using linear approximation. *Image and Vision Computing*, 12:487–498, 1994.

[21] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4):600–612, April 2004.

[22] G.-Q. Wei and G. Hirzinger. Learning shape from shading by a multilayer network. *Neural Networks, IEEE Transactions on*, 7(4):985–995, Jul 1996.

[23] P. Worthington and E. Hancock. New constraints on data-closeness and needle map consistency for shape-from-shading. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 21(12):1250–1267, December 1999.

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

Acknowledgements

L.T. was supported as part of an NSF REU program at UCF (IIS-0552807). M.T was supported by grant HM-15820810021 through the NGA NURI program.