Integrating Surface Normal Vectors using Fast Marching Method

Jeffrey Ho[1], Jongwoo Lim[2], Ming-Hsuan Yang[2], and David Kriegman[3]

[1] Department of CISE, University of Florida
Gainesville, FL USA
jho@cise.ufl.edu
[2] Honda Research Institute
Mountain View, CA USA
{jlim, myang}@honda-ri.com
[3] Department of Computer Science and Engineering, University of California
San Diego, CA USA
kriegman@cs.ucsd.edu

Abstract. Integration of surface normal vectors is a vital component in many shape reconstruction algorithms that require integrating surface normals to produce their final outputs, the depth values. In this paper, we introduce a fast and efficient method for computing the depth values from surface normal vectors. The method is based on solving the Eikonal equation using Fast Marching Method. We introduce two ideas. First, while it is not possible to solve for the depths Z directly using Fast Marching Method, we solve the Eikonal equation for a function W of the form $W = Z + \lambda f$. With appropriately chosen values for λ, we can ensure that the Eikonal equation for W can be solved using Fast Marching Method. Second, we solve for W in two stages with two different λ values, first in a small neighborhood of the given initial point with large λ, and then for the rest of the domain with a smaller λ. This step is needed because of the finite machine precision and rounding-off errors. The proposed method is very easy to implement, and we demonstrate experimentally that, with insignificant loss in precision, our method is considerably faster than the usual optimization method that uses conjugate gradient to minimize an error function.

1 Introduction

Many shape reconstruction algorithms in computer vision require integrating surface normal vectors. Reconstruction algorithms that use multi-view correspondences, such as structure from motion, generally recover the depth values directly from the pixel correspondences. However, algorithms that depend on exploiting illumination effects, such as photometric stereo and shape from shading, the depth values in general cannot be computed directly. Instead, under the usual Lambertian assumption, the normal vectors of the object's surface are recovered, and the depth values are obtained by integrating the surface normals. Successes abound in applying these techniques to important reconstruction problems in

computer vision, ranging from the human face reconstruction [1] to the more recent optical-flow based object reconstruction from video sequences [2]. And in all these successes, integration of normal vectors is an important part of the story. This paper proposes a new method for integrating surface normals, which, with insignificant loss in precision, is about two order of magnitude faster than the traditional algorithm that uses conjugate descent to minimize some error function. This improvement in performance is particularly noticeable on large-scale problems, with images containing up to two million pixels.

The normal integration problem can be stated very simply as follows. For an usual XY grid, we are given a normal vector $\mathbf{N} = \mathbf{N}(x, y)$ at each grid point (x, y). The task is then to recover a surface $\mathcal{S}$, represented by the height function $Z(x, y)$, such that $\mathbf{N}$ is a normal vector field of $\mathcal{S}$. A simple calculation shows that a normal vector of $\mathcal{S}$ at a point (x, y) is given by the formula,

$$\mathbf{N}(x, y) = (\frac{\partial Z}{\partial x}, \frac{\partial Z}{\partial x}, -1). \tag{1}$$

In the following discussion, we will adhere to the convention and denote quotients $\frac{N_x}{N_z}$ and $\frac{N_x}{N_z}$ by P and Q, respectively. Since the normal vector is only defined up to multiplication by a constant, for any normal vector $\mathbf{N}$, the ratios of its x and y components with its z-component are the partial derivatives of Z with respect to x and y, respectively. Namely,

$$\frac{\partial Z}{\partial x} = -\frac{N_x}{N_z} = -P, \tag{2}$$

$$\frac{\partial Z}{\partial y} = -\frac{N_y}{N_z} = -Q, \tag{3}$$

where N_x, N_y and N_z are the x, y and z components of $\mathbf{N}$. A straightforward way of solving this system of PDEs is then to minimize the following error function over the entire grid [3]:

$$\mathcal{E}(Z) = \sum_{i,j} (\frac{\partial Z}{\partial x} + \frac{N_x}{N_z})^2 + (\frac{\partial Z}{\partial y} + \frac{N_y}{N_z})^2. \tag{4}$$

The error function $\mathcal{E}$ is a quadratic function of its variables, $z_{i,j}$, the values of the function Z at the grid point (i, j). In principle, its global minimum can be determined by solving the K-by-K[4] linear system $Ax = b$ derived from the condition,

$$\nabla \mathcal{E} = 0. \tag{5}$$

While this is perfectly doable, it is definitely not recommended for a large system. For example, on an image of size 1401-by-1401, the dimension of the linear system above is roughly two millions. While A is sparse, it is still a daunting task to solve the linear system $Ax = b$ directly using, e.g. LU factorization,

[4] K is the number of grid points.

which has the complexity of $O(K^3)$. Therefore, conjugate gradient is often used to find a minimum of $\mathcal{E}$. The main problem with this approach is the speed of convergence. As is well-known, it depends on the initial point (some given height function Z) that starts the iteration as well as the conditioning of the matrix A. Of course, it also depends on the size of the problem. Not surprisingly, for large scale problems, the convergence of the conjugate gradient optimization of Equation 4 is often excruciatingly slow.

The other commonly used method for solving the normal integration problem is to transform the problem to the frequency domain [4]. Suppose P and Q have the following Fourier expansions:

$$P = \sum c_P(\omega_x, \omega_y) e^{i(\omega_x x + \omega_y y)},$$
$$Q = \sum c_Q(\omega_x, \omega_y) e^{i(\omega_x x + \omega_y y)},$$

where ω_x, ω_y are the fundamental frequencies. Then, a best surface (in the least square sense) is then given by the formula

$$Z = \sum c_(\omega_x, \omega_y) e^{i(\omega_x x + \omega_y y)},$$

where

$$c(\omega_x, \omega_y) = \frac{i\omega_x c_P(\omega_x, \omega_y) + i\omega_y c_Q(\omega_x, \omega_y)}{\omega_x^2 + \omega_y^2}$$

Fast Fourier Transform can be applied to efficiently solve the problem. However, as is well-known, FFT works well only with grids whose sizes are powers of 2, and there are also other problems associated with this approach.

In this paper, we propose a fast method for solving the normal integration problem. The algorithm is based on solving the Eikonal equation, and it uses the Fast Marching Method developed by Sethian and others for solving the Eikonal equation [5]. Our idea is as follows. While Fast Marching Method cannot be applied directly to solve for the height values Z, we can nevertheless try to solve for a function W of the form $W = Z + \lambda f$, where λ is a parameter and f is some known function. The idea is to find a pair of λ and f so that we can use Fast Marching Method to solve for the Eikonal equation for W. Even though the idea is simple, to the best of our knowledge, nothing similar has been reported in the computer vision literature before.

This paper is structured as follows. In the next section, we describe the proposed method of integration. In Section 3, we briefly review some related work, and we describe the similarities as well as disparities between the problem we solve here and the shape from shading problem. Experimental results comparing our method with the direct optimization of Equation 4 using conjugate gradient is reported in Section 4. The paper ends with a short summary and conclusion.

2 Integrating Surface Normals by Solving Eikonal Equation

It is well-known that the problem of integrating surface normals is intimately related to the solution of the Eikonal equation (e.g. [5]). Starting with the pair of equations in Equations 2 and 3, we have the following Eikonal equation:

$$\|\nabla Z\| = \sqrt{P^2 + Q^2}. \tag{6}$$

Therefore, a solution of the above equation is the desired height function Z. The Eikonal equation has appeared in various places in computer vision literature. For example, in many algorithms that use the level-set technique, Eikonal equation is often solved to produce a signed-distance function from the level set, and this re-initialization of the level-set function is a crucial step. Lately, there has been a considerable amount of interests in studying a modified Eikonal equation for solving the shape from shading problems [6]. Our approach here is also about solving the Eikonal equation.

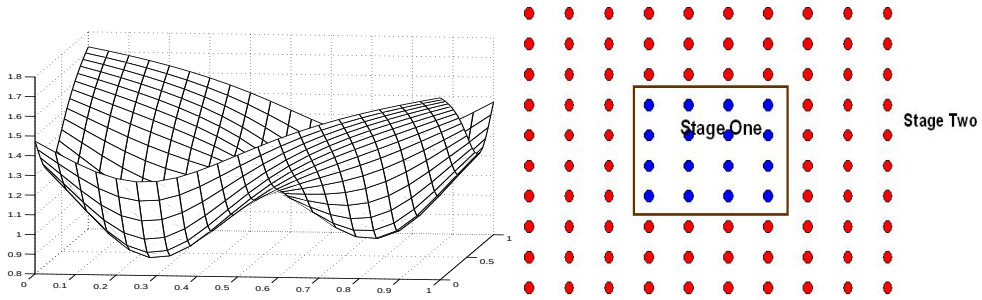

Fig. 1. Left: A function with two local minimums that cannot be recovered using Fast Marching Method. **Right**: We solve for W with two different λ values in two complementary regions.

The Fast Marching Method [5] provides a very efficient method to solve the Eikonal equation. Starting with an initial value at some given point u, it determines the Z value at every point by computing the time of arrival of an expanding wavefront. This method is very efficient, and it has the time complexity of roughly $\mathrm{O}(K \log K)$ with K the number of grid points, using an auxiliary heap structure for keeping track of the wavefront. In addition, it is also very easy to implement. Unfortunately, Fast Marching Method cannot be applied directly here. For one thing, the initial point u has to be the global minimum of Z and in general, this information is not available. Furthermore, it will also have a difficulty dealing with functions that have local minimums. For example, the simple function depicted in Figure 1(**Left**) with two local minimums cannot be recovered by a straightforward application of Fast Marching Method. A modified Fast Marching approach [5][7] is to determine the local minimums first, and

starting from these local minimums, every step extends reconstruction to higher depths and the entire reconstruction is then accomplished in one single pass. However, in our problem, we do not assume that we know the locations of these local minimums. In principle, one can detect the local minimums by determining the locations where $\partial Z/\partial x = \partial Z/\partial y = 0$, **and** the Hessian

$$H(x,y) = \begin{pmatrix} \frac{\partial^2 Z}{\partial^2 x} & \frac{\partial^2 Z}{\partial y \partial x} \\ \frac{\partial^2 Z}{\partial x \partial y} & \frac{\partial^2 Z}{\partial^2 y} \end{pmatrix} \tag{7}$$

is positive definite. However, with the noise present in the data as well as quantization effect, there is no guarantee on how accurately these local minimums can be located.

Instead, we propose to solve the Eikonal equation for a function W of the form

$$W = Z + \lambda f, \tag{8}$$

where λ is a constant and f is some function such that W is a function with one single global minimum at the initial point u and without any other critical points. In a way, the function f is here to cancel off any critical point of Z so that W is critical point free except at u. In particular, the level-set $W^{-1}(c)$ is always topologically the same for any value of c such that $W^{-1}(c)$ contains more than one point. Clearly, W can be solved using Fast Marching Method, and hence Z can be recovered from W if f and λ are known.

Since we are solving the height function Z over a finite domain, we can assume that both Z as well as its derivatives are bounded, $|Z| < c_1$ and $\|\nabla Z\| < c_2$. In practice, this is not a restrictive assumption since the surfaces been recovered by most shape reconstruction algorithms are often assumed to be smooth, and in many variational approaches [8], there is usually a smoothing term that minimizes the norm of ∇Z anyway. With this assumption in mind, we take f to be the squared-distance function (from the point $u = (u_x, u_y)$):

$$f = (x - u_x)^2 + (x - u_y)^2.$$

Without loss of generality, we assume that the initial point u is the origin in the following discussion. The derivatives of f vanish at the origin; therefore, u is not a critical point of $W = Z + \lambda f$ unless it is a critical point of Z. However, with sufficiently large λ, one can show that the critical points of $W = Z + \lambda f$ are all confined to a disk centered at origin with radius $r = q/2$, where q is the grid spacing, the distance between two neighboring grid points:

Lemma 1. *Suppose $\|\nabla Z\| < c_2$. If $\lambda > \frac{c_2}{2R}$ for some real number R, then $W = Z + \lambda f$ has no critical point outside of a disk $\mathcal{D}_r$ centered at the origin with radius R.*

The proof is trivial since in the region outside $\mathcal{D}_r$, $\|\nabla f\| > 2R$ and $\|\lambda \nabla f\| > 2R\lambda > c_2 > \|\nabla Z\|$. That is, $\nabla W = \nabla Z + \lambda \nabla f$ can never be zero outside of $\mathcal{D}_r$ since ∇Z and $\lambda \nabla f$ can never be the same. Therefore, in principle, we can choose a sufficiently large λ such that W only has critical points in the disk of

radius $q/2$. Since we are computing everything on the grid, these critical points will be invisible, and we can treat the origin as the only critical point of W. The constant c_2 can be determined in a single pass over the input datum P and Q. Note that since we know the derivatives of Z as well as those of f, the Eikonal equation for W is simply

$$\|\nabla W\| = \sqrt{(P + 2x)^2 + (Q + 2y)^2}. \tag{9}$$

Once W is computed, Z can be easily recovered.

While the approach above is mathematically valid, because of finite machine precision, large λ would have incurred large rounding-off errors for the Z values. The situation is particularly urgent for points far away from the origin, where the term λf would have been considerably larger than Z. In this region, we prefer a small λ, while in the region close to the origin, we can accommodate a larger value for λ. Our second idea is then to solve the problem in two stages. In the first stage, we solve for Z in a small neighborhood of the origin, e.g., in a disk of radius $R = 1$. In this region, we can use larger values for λ because the term $\lambda(x^2 + y^2)$ will in general still be manageable. In the second stage, we solve Z for the rest of the domain using the result from the first stage as the initial values. By the Lemma above, we can take λ to be $\frac{c_2}{2} + 1$. See Figure 1(**Right**). In our implementation, we choose the neighborhood $\mathcal{D}_r$ beforehand, and we fix it to be a window $\mathcal{W}$ of size, say 15-by-15, centered at the given point u. We take λ to be

$$\lambda = \max_{p=(x,y)\notin\mathcal{W}} \frac{\|\nabla Z\|}{2\|p\|} + 1 = \max_{p=(x,y)\notin\mathcal{W}} \frac{\sqrt{P^2 + Q^2}}{2\sqrt{x^2 + y^2}} + 1. \tag{10}$$

Again, λ can be determined in a single pass over the input datum P and Q.

The other way to solve for the Z values in a small neighborhood of the origin is to explicitly invert the linear system $Ax = b$. Since the neighborhood is supposed to be small, it makes sense to solve the system directly, and the inversion process is relatively cheap. In particularly, if we fix the size of this neighborhood $\mathcal{W}$, and because A depends only on the connectivity of $\mathcal{W}$, the LU factorization of A can be computed off-line and the online inversion of the linear system is then fast and effortless.

3 Comparison with Shaping from Shading Literature

One common method for solving the shape from shading problem is to solve an Eikonal equation of the form [5]:

$$\|Z\| = \sqrt{\frac{1}{I^2} - 1}. \tag{11}$$

Here we assume that the single light source is from the direction $(0, 0, 1)$, and the Lambertian object has uniform albedo of value 1. I above denotes the image

intensity value. Many papers have been devoted to solving this equation (e.g. [6]). We point out, however, that the major distinction between our problem of integrating surface normal vectors and the shape from shading problem formulated above is that in our case, we have the values for the x and y components of ∇Z, while in the shape from shading problem, only the magnitude of ∇Z is known, and it is related to the intensity value through the equation above.

It is precisely because we know the x and y components of ∇Z, we can solve the Eikonal equation for W in Equation 9 since the right hand side can be computed. An equation analogous Equation 9 is not available for the shape from shading problem. Therefore, more elaborated scheme has to be designed in order to solve the Eikonal equation efficiently.

Finally, we also mention one important fact about the comparison between using the proposed method and the direct minimization of Equation 4 using conjugate gradient descent. While Fast Marching Method is unquestionably efficient in solving the Eikonal equation, it is not an iterative process and therefore, there is no way to further improve the quality of the solution. Typically, as in the experiments reported in the next section, there will always be errors between the reconstructed depths and the true depths. In our experiment, the error is usually at most 1% of the true depth value. The point we try to make in this paper is that to reach the precision obtained via our method would usually require conjugate gradient to run as much as 200 times longer. However, being an iterative scheme, conjugate gradient can keep running until it reaches a global minimum (or within the given tolerance set by the machine precision and the user). Therefore, there are two ways to apply our proposed method in a normal integration problem. If the precision requirement is stringent (with relative error $< 10^{-3}$), one can use the proposed method to quickly obtain an initial surface estimate, and feed this result into an efficient optimization method to yield a more precise result. On the other hand, if the precision requirement is not too demanding, the output of our method will usually be sufficient.

4 Experiments and Results

In this section, we report our experimental results. All experiments reported below were run on a 3.19 GHz DELL Pentium desktop computer with 2.00 GB of RAM running Windows XP. We implemented the proposed method using a simple C++ implementation without any optimization except a heap structure for keeping track of the front points and their neighbors. We compare the performance of the proposed method with the standard conjugate gradient minimization of Equation 4 [5]. The implementation of the conjugate gradient descent is taken straight out of [9]. Except for the Brent line minimization, no further optimization of the code has been implemented.

[5] The standard conjugate gradient without line search usually fails to converge to the precision we required. Therefore, we include the line minimization to ensure that the iterative process will converge to the required precision.

Below, we provide two types of experiments. In the second group of experiments, we work with the (noisy) normal vectors of a human face estimated using the photometric stereo algorithm [1]. In the first group of experiments, which is our main focus, we work with surfaces with known depths and normal vectors. The goal is to compare the speed of our method and that of the conjugate gradient descent under the same precision requirement. For each of the four functions $Z = Z(x, y)$ given below, we compute the normals of the surface represented by Z using Equation 1. The estimated depth value Z' is computed using both methods at each point, and also the relative error

$$\epsilon = \frac{|Z - Z'|}{|Z|}.$$

The mean, median and also the standard deviation of the relative errors are computed for each function, and they serve as the measurements used to compare both methods.

4.1 Experiment with Simulated Data

In this group of experiments, we study the performance of our method for surfaces with known depths and normals. We run the experiments on a grid of size 1401×1401, and there are roughly two million grid points. In all experiments, it takes less than three seconds (2.677 to be exact) for our method to finish. Note that the speed of our method is independent of the value of λ as well as the chosen initial starting point. We compute the mean, median and standard deviation of the relative errors of the reconstruction result given by our method. We then run the conjugate gradient (CG) with sufficiently many iterations in order to reach the same precision requirement. In the experiments reported below, CG usually takes about 250 to 350 iterations to converge to the required precision, and this translates into roughly from 475 to 700 seconds. Averagely, our method is about 200 times faster than the conjugate gradient method.

Sphere For the first experiment, we look at the simplest case of a sphere,

$$Z = \sqrt{1.5^2 - x^2 - y^2}$$

over the domain $\mathcal{D} \equiv \{(x, y)| - 0.7 \leq x, y \leq 0.7\}$. In this example, we take the domain $\mathcal{D}_r$ to be the disk with radius $r = 7$ grid points. For the first experiment, the initial starting point is chosen at the center of the grid, the apex of the sphere over the domain $\mathcal{D}$. We pick the optimal value of $\lambda = 6$ determined by Equation 9. The mean, median and the standard deviation of the relative errors with this λ setting are 0.0046, 0.0045, and 0.0015, respectively, which is sufficiently accurate for many applications.

Next, we vary the value of λ from 0 to 100. The mean, median and the stand derivation for these values of λ are plotted in Figure 2. The optimal λ value of 6 is very close to the empirical optimal value of $\lambda = 4$, which gives the mean, median,

standard deviation of 0.0042, 0.0042, and 0.0015, respectively. Note that with $\lambda = 0$, which corresponds to a direct application of Fast Marching Method to solve the Eikonal equation, the result is, as expected, completely incorrect. Also as expected, as the value of λ increases, the rounding-off errors start creeping in and accumulating, the reconstruction result begins to deteriorate. However, even with the relatively large value of $\lambda = 60$, the median and mean of the relative error is still below 3% with standard deviation less than 2%.

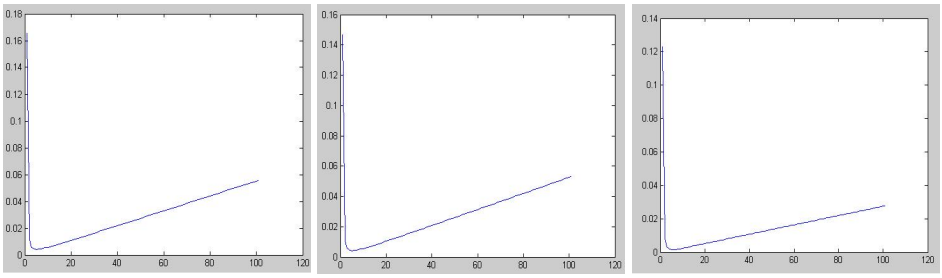

Fig. 2. From Left to Right: Plots of the mean, median and standard deviation of the relative errors of the reconstruction results for the sphere with λ ranging from 0 to 100.

In Figure 3, we show the reconstruction results using $\lambda = 0, 6, 100$. Clearly, the reconstruction result for $\lambda = 0$ is completely incorrect.

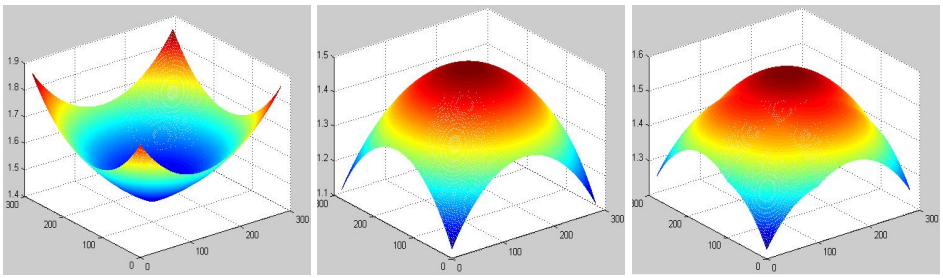

Fig. 3. From Left to Right: Reconstruction results for the sphere using $\lambda = 0, 6$ and 100, respectively. The images are colored-coded according to depth values.

Monkey Saddle While we have passed the rudimentary test using sphere, the next example, which is a little more challenging, uses the function

$$Z = x(x^2 - 3y^2) + 3.$$

Instead of the global maximum in the example above, the origin is now a saddle point of the function Z. We run the experiment over the same range of λ, from

0 to 100. The mean, median and the stand derivation of the relative errors for these values of λ are plotted in Figure 4. Again, we observe the similar pattern as above that when $\lambda = 0$, the reconstruction result is completely incorrect. Furthermore, the quality of the reconstruction, as measured by the mean, median and standard deviation of the relative errors, deteriorates as λ increases. In this example, we have used $\lambda = 12$ and it is close to the empirical optimal value of $\lambda = 18$.

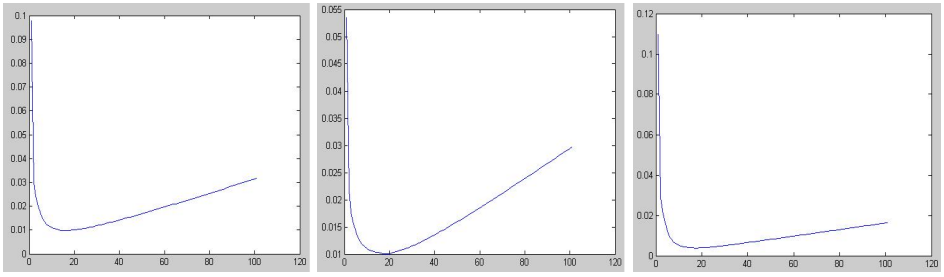

Fig. 4. From Left to Right: Plots of the mean, median and standard deviation of the relative errors of the reconstruction results for the Monkey Saddle with λ ranging from 0 to 100.

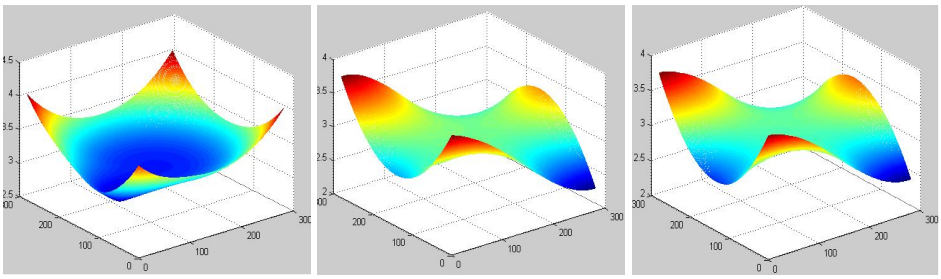

Fig. 5. From Left to Right: Reconstruction results for the Monkey Saddle using $\lambda = 0, 12$ and 100, respectively.

Sinusoidal Function and Gaussian Figures 6 and 7 display the reconstruction results for the following two functions:

$$Z = \sin(2\pi(x^2 + y^2)) + 3 \tag{12}$$

$$Z = e^{-x^2-y^2} + 10 \tag{13}$$

For the gaussian exponential function, again, we see that the reconstruction result for $\lambda = 0$ is incorrect. For the function $Z = \sin(2\pi(x^2 + y^2)) + 3$, instead of displaying the reconstruction result for $\lambda = 0$, we show the result with $\lambda = 4$, which is clearly incorrect and incomplete.

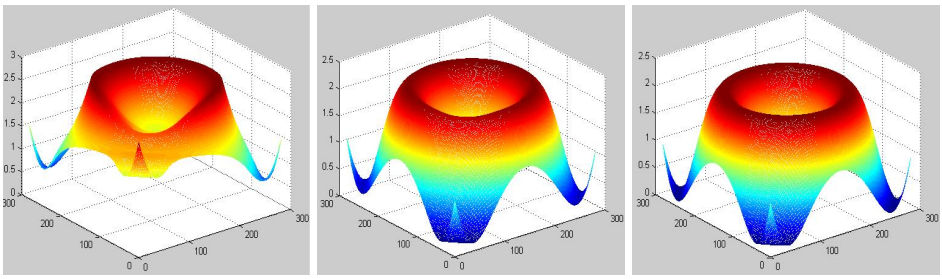

Fig. 6. From Left to Right: Reconstruction results for the function $Z = \sin(2\pi(x^2 + y^2)) + 3$ using $\lambda = 4, 30$ and 100, respectively.

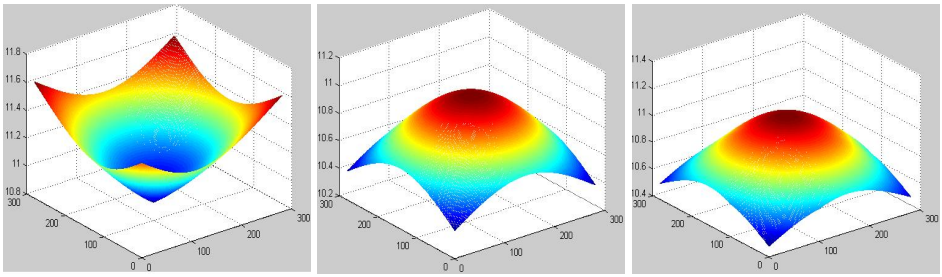

Fig. 7. From Left to Right: Reconstruction results for $Z = e^{-x^2-y^2} + 10$ using $\lambda = 0, 8$ and 100, respectively.

4.2 Experiment with Real Data

In this experiment, we work with the normal vectors provided in the Yale Face Database B [1]. Images of each of the ten individuals in the database were taken under different illumination conditions. The normal vectors are estimated using the photometric stereo algorithm of [10]. Figure 8 shows the reconstruction result using our method for one individual in the database. Since the image here is of size 168×192, which is considerably smaller than the ones we used above, it takes less then 0.02 second for our method to complete the integration.

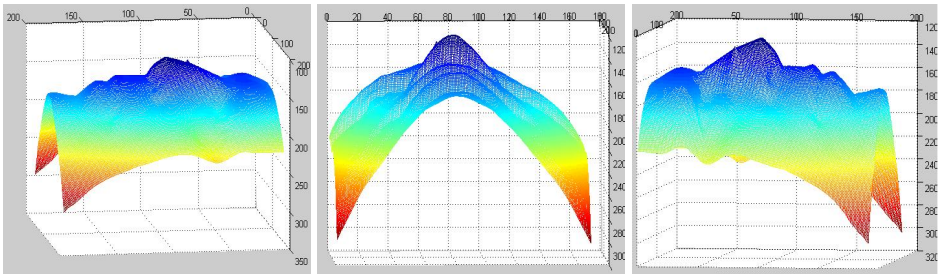

Fig. 8. From Left to Right: Three views of the reconstruction result of one individual in the Yale Face Database B.

5 Conclusion

In this paper, we have presented a method for computing depth values from surface normal vectors. The proposed method is based on the Fast Marching Method for solving the Eikonal equation. Our main contribution is the observation that while we cannot apply Fast Marching Method directly to solve the Eikonal equation for the unknown depth Z directly, we can solve the Eikonal equation for a function W, which is the sum of the unknown depth plus and some function. The idea is that W is a function with one critical point and Fast Marching Method can be applied to solve W quickly, and hence Z recovered. Because of the finite machine precision and rounding-off errors, we are forced to solve W in two stages, first in a small neighborhood containing the given initial point and then solve W for the rest of the domain. We have presented several different experiments with synthetic examples, which allow us to examine precisely several aspects of our algorithm. In all examples, we have demonstrated that given the same precision requirement, the proposed method is considerably faster than the old method based on conjugate gradient descent. Since surface normal integration is an important component in many shape reconstruction algorithms, we believe that the results presented in this paper will be of interest to a sizable portion of the computer vision community.

6 Acknowledgements

This work was partially supported by NSF IIS-0308185, NSF EIA-0224431, NSF CCR 00-86094, University of Florida and the Honda Research Institute.

References

1. Georghiades, A., Kriegman, D., Belhumeur, P.: From few to many: Generative models for recognition under variable pose and illumination. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 643–660

2. L. Zhang, B. Curless, A.H., Seitz, S.: Shape and motion under varying illumination: Unifying structure from motion, photometric stereo, and multi-view stereo. In: Proc. Int. Conf. on Computer Vision. (2003) 618–625
3. Horn, B.K.P., Brook, M.J.: Shape from Shading. MIT Press (1997)
4. Frankot, R., Chellappa, R.: A method of enforcing integrability in shape from shading algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence **10** (1988) 439–451
5. Sethian, J.A.: Level Set Methods and Fast Marching Methods. Cambridge Press (1996)
6. Prado, E., Faugeras, O.: Shape from shading: A well-posed problem ? In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog. (2005) 158–164
7. A. Tankus, N.S., Yeshurun, H.: Perspective shape from shading by fast marching. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog. (2004) 618–625
8. Trucco, E., Verri, A.: Introductory Techniques for 3D Computer Vision. Prentice Hall (1998)
9. W. H. Press, S. A. Teukolsky, W.T.V., Flannery, B.P.: Numerical Recipes in C, Second Edition. Cambridge Press (1992)
10. Yuille, A., Snow, D.: Shape and albedo from multiple images using integrability. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog. (1997) 158–164