

Shape From Shading Using Linear Approximation *

Ping-Sing Tsai and Mubarak Shah

Department of Computer Science

University of Central Florida

Orlando, FL 32816

Abstract

In this paper, we present an extremely simple algorithm for shape from shading, which can be implemented in 25 lines of C code ¹. The algorithm is very fast, taking .2 seconds on a Sun SparcStation-1 for a 128×128 image, and is purely local and highly parallelizable (parallel implementation included). In our approach, we employ a linear approximation of the reflectance function, as used by others. However, the major difference is that we first use the discrete approximations for surface normal, p and q , using finite differences, and then linearize the reflectance function in depth, $Z(x,y)$, instead of p and q . The algorithm has been tested on several synthetic and real images of both Lambertian and specular surfaces, and good results have been obtained.

*The research reported here was supported by the Florida HiTech Council under grant 65 02 731, and the National Science Foundation under grant number CDA 9100898.

¹C code and some images can be obtained by anonymous ftp from eustis@cs.ucf.edu under the directory /pub/shading.

Contents

1	Introduction	3
2	Shape from Shading	5
2.1	Lambertian Surfaces	5
2.2	Specular Surfaces	7
3	Comparison with Pentland's method	9
4	Convergence	10
5	Implementation Details	13
6	Experiments	16
6.1	Results for Lambertian Surfaces	16
6.2	Result for Specular Surfaces	21
7	Parallel Implementation	21
8	Conclusions	24

1 Introduction

Shape from shading deals with the recovery of 3D shape from a single monocular image. The recovered shape can be in terms of depth Z , the surface normal (n_x, n_y, n_z) , or surface gradient (p, q) . There are two main classes of algorithms for computing shape from a shaded image: global methods, and local methods. In the global methods, the shape is recovered by minimizing some cost function involving constraints such as smoothness. In these approaches the variational calculus technique is used to iteratively compute the shape, which is globally consistent. In the local methods, the shape is recovered by using local constraints about the surface being recovered, or about the reflectance map being used. The common assumptions about surface shape and reflectance are that the surface is locally spherical, and the reflectance map is linear in surface shape. The local methods, in general, are simple, but only provide an approximate shape. The global methods, on the other hand, are more complex, but provide very close to accurate shape. A representative method of global approaches is used by Horn [3]. His algorithm simultaneously computes both the depth and the surface gradient. He combines three constraints: the brightness constraint, integrability constraint, and gradient smoothness constraint. The local shape from shading algorithms are by Pentland [8], and Lee and Rosenfeld [4].

In shape from shading algorithms it is assumed that the reflectance map is given, or its form is known. Since images of most surfaces in the real world can be approximated by Lambertian reflectance, the majority of shape from shading methods use the Lambertian reflectance model. The important parameters in Lambertian reflectance are albedo and illuminant directions. Commonly, the albedo is assumed to be constant. There are several methods for computing the light source direction originated by Pentland [7]. These methods assume the surface to be locally spherical. Recently, Chellappa and Zheng [12] proposed two methods for computing the light source direction, local voting method and contour-based method. These methods are more robust and accurate for synthetic and real images, and more consistent with different bands of real color images, different subimages of a scene, or images of different resolutions. The authors also provide a good comparison between the methods of Pentland, Lee and Rosenfeld, and their own method.

Chellappa and Zheng [12] proposed a global algorithm based on constrained function

optimization. A typical cost function (e.g. one used by Horn) involves an intensity constraint, regularization constraint, and integrability constraint. The key idea in Chellappa and Zheng’s approach is that they introduce a new cost function which does not use the quadratic regularization term. Instead, they require the gradients of the reconstructed image to be close to the gradients of the input image. Chellappa and Zheng also use the linear approximation for the reflectance function around the surface normal (p, q) by taking the Taylor series expansion up to the first-order terms, similar to Pentland.

Oliensis [6] has suggested that the smoothness term in the objective function is unnecessary for images containing singular points, i.e. maximally bright points. He believes that smoothness will distort the surface reconstruction. Recently, Bichsel and Pentland [1] have presented a simple algorithm along the lines of Oliensis. The algorithm is based on a minimum downhill principle which guarantees continuous surfaces and stable results.

Lee and Kuo [5] recently proposed an algorithm which is based on a triangular element surface model and the linear approximation of the reflectance map. They approximate a smooth surface by the union of triangular surface patches which can be expressed as a linear combination of a set of nodal basis functions. The depth value was computed by minimizing a quadratic cost functional of brightness error. Their method does not require any integrability constraints or assumptions about boundary conditions.

Pentland [8] proposed a local algorithm based on the linearity of the reflectance map in the surface gradient (p, q) , which greatly simplifies the shape from shading problem. Later, Pentland [9] presented an extension of his linear model to quadratic surfaces using photometric motion for extracting shape and reflectance.

We believe that the linearity of the reflectance map in the depth Z , instead of in p and q , is more appropriate in some cases, and its use results in a better depth map. In this paper we present a new method for computing depth from a single shaded image. In our approach, we employ the discrete approximations for p and q using finite differences, and linearize the reflectance in $Z(x, y)$. Our method is faster, since each operation is purely local. In addition, it gives good results for the spherical surfaces, unlike other linear methods. Moreover, our method is more general, and can be applied to any reflectance function. For instance, in this paper we describe an extension of this method to the specular surface.

The organization of the rest of the paper is as follows. The next section deals with the

main part of this paper where we describe our method for shape from shading. We consider both the Lambertian and specular surfaces. In section three we present a comparison between our method and Pentland’s method, and highlight the advantages of our method. Next, the issue of convergence of a solution is addressed in section four. Our algorithm is extremely simple to implement, except in some special cases. We discuss those cases and provide a general algorithm in Section 5. The method has been tested extensively on a large set of real and synthetic images of both Lambertian and specular surfaces. Our experiments are summarized in Section 6.

2 Shape from Shading

2.1 Lambertian Surfaces

The reflectance function for the Lambertian surfaces is modeled as follows:

$$E(x, y) = R(p, q) \quad (1)$$

$$\begin{aligned} &= \frac{1 + pp_s + qq_s}{\sqrt{1 + p^2 + q^2} \sqrt{1 + p_s^2 + q_s^2}} \\ &= \frac{\cos \sigma + p \cos \tau \sin \sigma + q \sin \tau \sin \sigma}{\sqrt{1 + p^2 + q^2}} \end{aligned} \quad (2)$$

where $E(x, y)$ is gray level at pixel (x, y) , $p = \frac{\partial Z}{\partial x}$, $q = \frac{\partial Z}{\partial y}$, $p_s = \frac{\cos \tau \sin \sigma}{\cos \sigma}$, $q_s = \frac{\sin \tau \sin \sigma}{\cos \sigma}$, τ is the tilt of the illuminant and σ is the slant of the illuminant. Using the following discrete approximations for p and q

$$p = \frac{\partial Z}{\partial x} = Z(x, y) - Z(x - 1, y) \quad (3)$$

$$q = \frac{\partial Z}{\partial y} = Z(x, y) - Z(x, y - 1), \quad (4)$$

the above reflectance equation can be rewritten as:

$$\begin{aligned} 0 &= f(E(x, y), Z(x, y), Z(x - 1, y), Z(x, y - 1)) \\ &= E(x, y) - R(Z(x, y) - Z(x - 1, y), Z(x, y) - Z(x, y - 1)). \end{aligned} \quad (5)$$

For a fixed point (x, y) and a given image E , a linear approximation (Taylor series expansion up through the first order terms) of the function f (equation 5) about a given depth map

Z^{n-1} is

$$\begin{aligned}
0 &= f(E(x, y), Z(x, y), Z(x-1, y), Z(x, y-1)) \\
&\approx f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&\quad + (Z(x, y) - Z^{n-1}(x, y)) \frac{\partial}{\partial Z(x, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&\quad + (Z(x-1, y) - Z^{n-1}(x-1, y)) \frac{\partial}{\partial Z(x-1, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&\quad + (Z(x, y-1) - Z^{n-1}(x, y-1)) \frac{\partial}{\partial Z(x, y-1)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1))
\end{aligned} \tag{6}$$

The above equation can be written as follow:

$$\begin{aligned}
&\frac{\partial}{\partial Z(x, y-1)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) * Z(x, y-1) \\
&+ \frac{\partial}{\partial Z(x-1, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) * Z(x-1, y) \\
&+ \frac{\partial}{\partial Z(x, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) * Z(x, y) \\
= &- f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&+ Z^{n-1}(x, y) * \frac{\partial}{\partial Z(x, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&+ Z^{n-1}(x-1, y) * \frac{\partial}{\partial Z(x-1, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&+ Z^{n-1}(x, y-1) * \frac{\partial}{\partial Z(x, y-1)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)),
\end{aligned}$$

or in vector form as follow:

$$(0, \dots, a_{x, y-1}, 0, \dots, a_{x-1, y}, a_{x, y}, 0, \dots) \begin{pmatrix} Z_{1,1} \\ \vdots \\ Z_{x,y} \\ \vdots \\ Z_{N,N} \end{pmatrix} = b_{x,y}, \tag{7}$$

where $a_{x,y} = \frac{\partial}{\partial Z(x,y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1))$ and

$$\begin{aligned}
b_{x,y} = &- f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&+ Z^{n-1}(x, y) * \frac{\partial}{\partial Z(x, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&+ Z^{n-1}(x-1, y) * \frac{\partial}{\partial Z(x-1, y)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)) \\
&+ Z^{n-1}(x, y-1) * \frac{\partial}{\partial Z(x, y-1)} f(E(x, y), Z^{n-1}(x, y), Z^{n-1}(x-1, y), Z^{n-1}(x, y-1)).
\end{aligned}$$

For a $N \times N$ image, there are N^2 such equations which will form a linear systems, $A \cdot Z = B$, where A is a $N^2 \times N^2$ matrix, Z and B are $N^2 \times 1$ vectors. This linear system is difficult to solve directly, since it will involve the inverse of a huge matrix, A . However, it can be solved easily using the Jacobi iterative method. Now, let us look carefully inside the Jacobi iterative method. For a given initial approximation Z^0 , each depth value is solved sequentially in a iteration. For example, the depth value $Z(x, y)$ at the n^{th} iteration can be solved using the previous estimates, $Z^{n-1}(i, j)$, for all the $Z(i, j)$ with $i \neq x$ and $j \neq y$. When $Z^{n-1}(x-1, y)$ and $Z^{n-1}(x, y-1)$ are respectively substituted for $Z(x-1, y)$ and $Z(x, y-1)$ in equation 6 the third and fourth terms on the right hand side vanish. Therefore, equation 6 reduces to surprisingly simple form given in the following equation:

$$\begin{aligned} 0 &= f(Z(x, y)) \\ &\approx f(Z^{n-1}(x, y)) + (Z(x, y) - Z^{n-1}(x, y)) \frac{d}{dZ(x, y)} f(Z^{n-1}(x, y)). \end{aligned} \quad (8)$$

Then for $Z(x, y) = Z^n(x, y)$, the depth map at the n -th iteration, can be solved directly as follow:

$$Z^n(x, y) = Z^{n-1}(x, y) + \frac{-f(Z^{n-1}(x, y))}{\frac{d}{dZ(x, y)} f(Z^{n-1}(x, y))} \quad (9)$$

where

$$\frac{df(Z^{n-1}(x, y))}{dZ(x, y)} = -1 * \left(\frac{(p_s + q_s)}{\sqrt{p^2 + q^2 + 1} \sqrt{p_s^2 + q_s^2 + 1}} - \frac{(p + q)(pp_s + qq_s + 1)}{\sqrt{(p^2 + q^2 + 1)^3} \sqrt{p_s^2 + q_s^2 + 1}} \right). \quad (10)$$

Now, assuming the initial estimate of $Z^0(x, y) = 0$ for all pixels, the depth map can be iteratively refined using Equation 9. We have observed that application of Gaussian smoothing to the final depth map $Z^n(x, y)$ results in a much smoother depth map.

2.2 Specular Surfaces

Specularity only happens when the incident angle of the light source is equal to the reflective angle. It is formed by two components: the specular spike, and the specular lobe. The specular spike is zero in all directions except for a very narrow range around the direction of specular reflection. The specular lobe spreads around the direction of specular reflection.

The simplest model for specular reflection is described by the delta function as follows:

$$I_S = B\delta(\theta_s - 2\theta_n),$$

where I_S is the specular brightness, B is the strength of the specular component, θ_s is the angle between the light source direction and the viewing direction, and θ_n is the angle between the surface normal and the viewing direction. This model assumes that the highlight caused by specular reflection is only a single point. However, in real life, this assumption is not true.

Another model is developed by Phong [10]. It represents the specular component of reflection as powers of the cosine of the angle between the perfect specular direction and the viewing direction. This model is capable of predicting specularities which extend beyond a single point. However, the parameters in this model have no physical meaning.

A more refined model, the Torrance-Sparrow model [11], assumes that a surface is composed of small, randomly oriented, mirror-like facets. It describes the specular brightness as a product of four components: energy of incident light, Fresnel coefficient, facet orientation distribution function, and geometrical attenuation factor adjusted for foreshortening. On the basis of the Torrance-Sparrow model, Healey and Binford [2] came up with a simplified model by using the Gaussian distribution as the facet orientation function, and considering the other components as constants. It can be described as:

$$I_S = K e^{-\left(\frac{\alpha}{m}\right)^2}$$

where I_S is the specular brightness, K is a constant, α is the angle between the surface normal and the bisector of viewing direction and source direction ($\alpha = \cos^{-1}(N \cdot H)$), m indicates the surface roughness, H is the bisector of the light source direction and the view direction, and N is the surface normal ($N = \frac{(p,q,1)}{\sqrt{p^2+q^2+1}}$).

The reflectance function for the specular surface using the terminology in equation (1) can be modeled as follows:

$$E(x, y) = R(p, q) = I_S = K e^{-\left(\frac{\cos^{-1}(N \cdot H)}{m}\right)^2}.$$

Using the discrete surface normal as before, and applying the same technique as in the case of Lambertian, we have

$$\begin{aligned} 0 &= f(Z(x, y)) \\ &= E(x, y) - R(Z(x, y) - Z(x - 1, y), Z(x, y) - Z(x, y - 1)) \\ &= E(x, y) - I_S \end{aligned}$$

$$\approx f(Z^{n-1}(x, y)) + (Z(x, y) - Z^{n-1}(x, y)) \frac{df}{dZ(x, y)}(Z^{n-1}(x, y)), \quad (11)$$

where

$$\begin{aligned} \frac{df(Z^{n-1}(x, y))}{dZ(x, y)} &= -2.0 * K * e^{-\left(\frac{\cos^{-1}(N \cdot H)}{m}\right)^2} * \frac{\cos^{-1}(N \cdot H)}{m^2 * \sqrt{1 - (N \cdot H)^2}} \\ &* ((H_x + H_y + H_z) - \frac{(pH_x + qH_y + H_z) * (p + q)}{p^2 + q^2 + 1}) * (p^2 + q^2 + 1)^{-1/2}. \end{aligned}$$

Then the depth information can be recovered by the above formula with function f , as in the case of Lambertian.

3 Comparison with Pentland's method

Our method is similar to Pentland's [8] linear shape from shading method in some aspects; therefore, we will compare these two methods here. Pentland uses the linear approximation of the reflectance map in p and q . By taking the Taylor series expansion of the reflectance function R , given in equation (1), about $p = p_0$, $q = q_0$, up through the first order terms, we have

$$E(x, y) = R(p_0, q_0) + (p - p_0) \frac{\partial R}{\partial p}(p_0, q_0) + (q - q_0) \frac{\partial R}{\partial q}(p_0, q_0). \quad (12)$$

For Lambertian reflectance, the above equation at $p_0 = q_0 = 0$, reduces to

$$E(x, y) = \cos \sigma + p \cos \tau \sin \sigma + q \sin \tau \sin \sigma.$$

Next, Pentland takes the Fourier transform of both sides of the equation. Since the first term on the right is a DC term, it can be dropped. Using the identities:

$$\frac{\partial}{\partial x} Z(x, y) \longleftrightarrow F_Z(\omega_1, \omega_2)(-i\omega_1) \quad (13)$$

$$\frac{\partial}{\partial y} Z(x, y) \longleftrightarrow F_Z(\omega_1, \omega_2)(-i\omega_2), \quad (14)$$

where F_Z is the Fourier transform of $Z(x, y)$, we get,

$$F_E = F_Z(\omega_1, \omega_2)(-i\omega_1) \cos \tau \sin \sigma + F_Z(\omega_1, \omega_2)(-i\omega_2) \sin \tau \sin \sigma,$$

where F_E is the Fourier transform of the image $E(x, y)$. The depth map $Z(x, y)$ can be computed by rearranging the terms in the above equation, and then taking the inverse Fourier transform.

The major difference between Pentland’s method and our method is that instead of linearizing the reflectance in p and q , we use the discrete approximations for p and q in terms of Z , and then linearize the reflectance in $Z(x, y)$. In this way, we have the following advantages.

First, we feel that the linearization of reflectance in Z is better than the linearization in p and q . For instance, it produces a good depth estimate for the spherical surface as compared to Pentland’s method. Figure 1.a shows the gray level image of sphere without using any approximation. Figure 1.b shows the image generated using linear approximation of reflectance map in Z , Figure 1.c shows the histogram of difference in gray levels in 1.a and 1.b, and Figure 1.d shows the reconstructed depth by our method.

Second, when the light source direction and the viewing direction are similar (images with central illumination), as pointed out by Pentland, the quadratic terms of the surface normal (p^2 and q^2) will become dominating in the reflectance function, and the Fourier transforms of p^2 and q^2 will have a doubling effect in the frequency domain. Since we do not use the Fourier transform, we do not have any frequency doubling effect, and our method is more general as it can apply to both low-angle illumination and central illumination.

Third, note that the Fourier components exactly perpendicular to the illuminant cannot be seen in the image data, and must be obtained from the boundary conditions, or simply be set to zero. In our method, the depth is obtained from the intensity domain instead of the Fourier domain; Therefore, no boundary conditions are required.

Another advantage of our method is that, computationally, it is very simple. Each operation is purely local, hence the method is highly parallelizable. In Pentland’s method one needs to compute the Fourier and inverse Fourier transform of the whole image, which is time-consuming.

4 Convergence

Basically, our iterative algorithm is a form of the Newton-Raphson method. It is well known that the Newton-Raphson method converges quadratically when provided with a sufficiently accurate initial approximation. Generally speaking, a nonlinear system, $F(x) = 0$, where x is a vector of n unknowns, can be solved using Newton’s method for nonlinear systems.

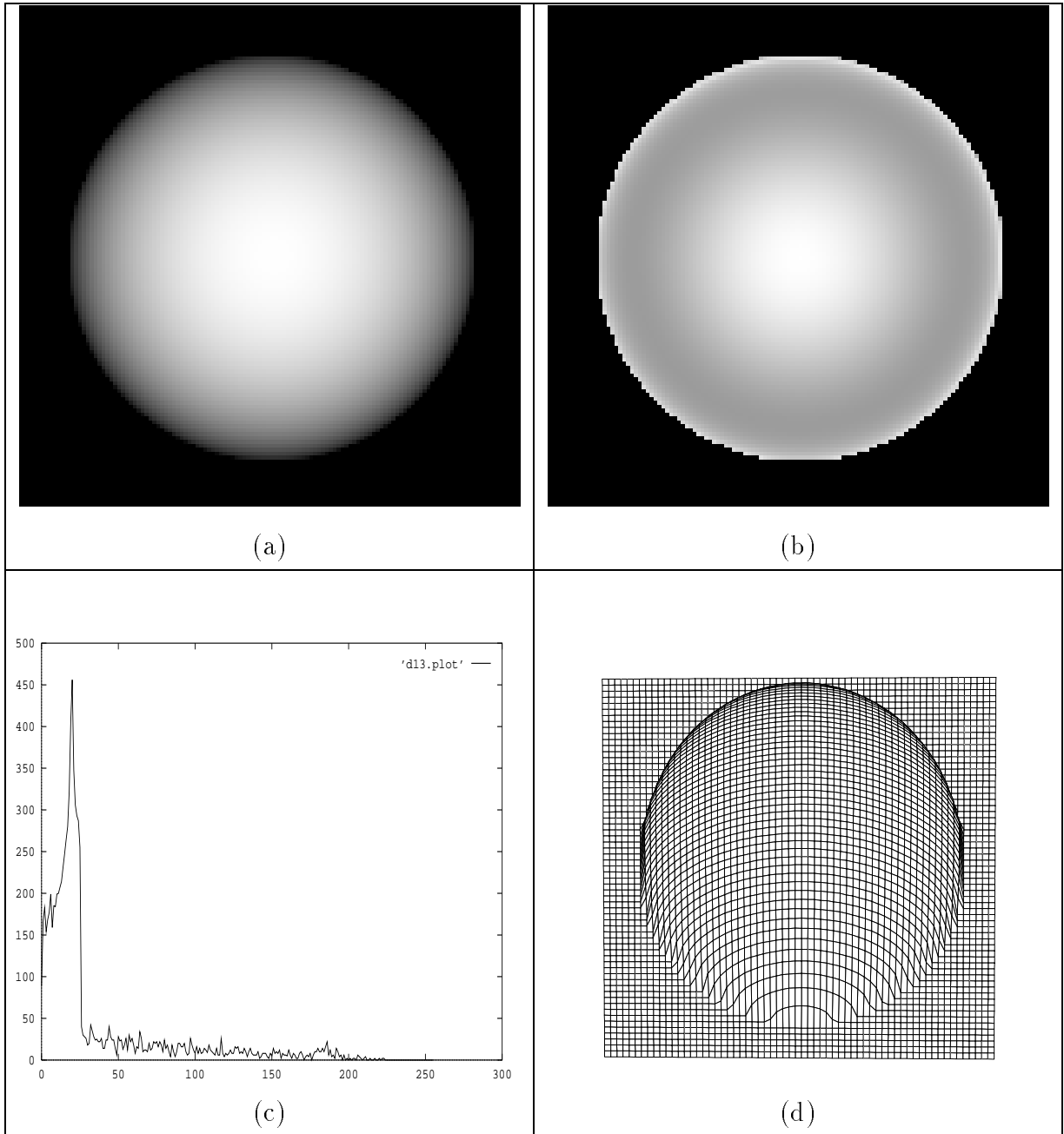


Figure 1: The results for Sphere Image using our method. (a) Gray level image without any approximation. The light source direction is $(0.01, 0.01, 1)$. (b) Gray level image using linearity in Z . (c) Histogram of difference in (a) and (b). (d) A 3-D plot of the depth map computed by our algorithm.

Newton's method is a functional iteration procedure which evolves from selecting x^0 and generating, for $k \geq 1$,

$$x^k = x^{k-1} - J(x^{k-1})^{-1}F(x^{k-1}), \quad (15)$$

where the matrix $J(x)$ is defined by

$$J(x) = \begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \frac{\partial f_2(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_2} & \cdots & \frac{\partial f_2(x)}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_n(x)}{\partial x_1} & \frac{\partial f_n(x)}{\partial x_2} & \cdots & \frac{\partial f_n(x)}{\partial x_n} \end{bmatrix},$$

and is called the Jacobian matrix. This method is generally expected to give quadratic convergence, provided that a sufficiently accurate starting value is known, and $J(x)^{-1}$ exists. It involves solving a linear system. This can be done by using the Jacobi iteration method, which is efficient in terms of both computer storage and computational times for a large system with a high percentage of zero entries.

Equation 6 is a nonlinear equation in three unknowns. We have a nonlinear system of N^2 such equations with N^2 unknown $Z(x, y)$ for a $N \times N$ image, where $Z(x, y)$ is the depth value at a point (x, y) . The way we solve it using linear approximation is really a form of Newton's method for a large nonlinear system with a high percentage of zero entries. As mentioned before, Newton's method is generally expected to converge quadratically when provided with a sufficiently accurate initial approximation. In our case, without any prior knowledge about the input image, the best initial estimation of depth, $Z(x, y)$, for each pixel is zero

In order to show that our algorithm converges to the correct answer and to evaluate its performance, we need to choose an error measure and test images. Horn [3] discusses a number of possible error measures. Here, we use the average magnitude of the gradient error $|p^n - \hat{p}| + |q^n - \hat{q}|$ as the error measure, where (p^n, q^n) is the estimated surface normal after n iterations, and (\hat{p}, \hat{q}) is the true surface normal. Since we do not have the true surface normal for the real images, we have performed the convergence tests on two synthetic images, Sphere and Mozart.

In Figure 2 the error-iteration curve for the sphere image is shown. We can clearly see that the average error of the surface normal is decreasing as the number of iterations increase.

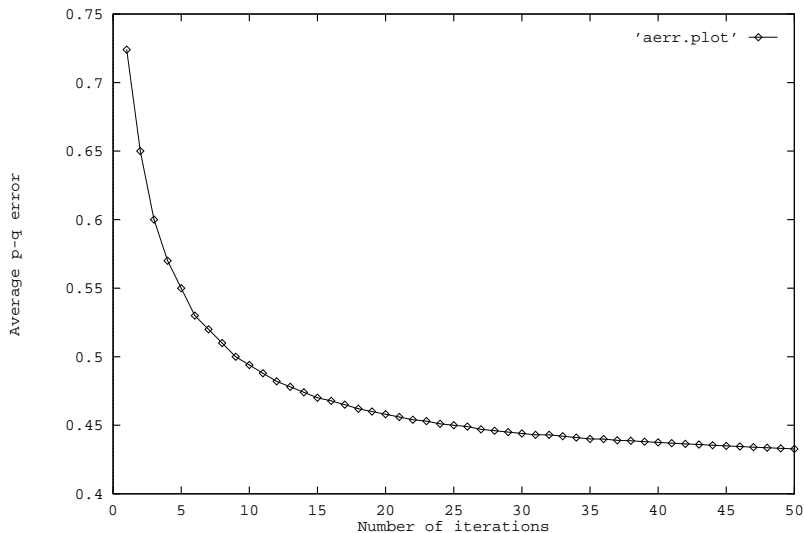


Figure 2: The error-iteration curve for the sphere image

The results for the Mozart image are shown in Figure 3. The true depth map shown in Figure 3.(a) was obtained by a laser range finder from the University of Southern California. The gray level image generated from the true depth map using a light source of $(0.01, 0.01, 1)$ is shown in Figure 3.(b). The gray level image generated from the estimated depth map by our method and using the same light source is shown in Figure 3.(c). Figure 3.(d) shows the error-iteration curve for the first 20 iterations.

5 Implementation Details

Our iterative algorithm can be implemented very straightforwardly. Assuming the initial estimate of $Z^0(x, y) = 0$ for all pixels, we only need to compute the function $f(Z^{n-1}(x, y))$, and the first derivative of the function $f'(Z^{n-1}(x, y))$ at each iteration. The formula in equation (9) will refine the depth map at each step. However, recall that the first derivative of the function $f'(Z^{n-1}(x, y))$ in Equation (10) guarantees a nonzero value only for the first step. Depending on the surface shape of the object, Equation (10) could become zero, which cause division by zero in equation (9). For example, when the surface normal is directly facing the light source ($p = p_s$ and $q = q_s$), or when $p = q$ and $p + q = p_s + q_s$, the derivative of the function $f'(Z(x, y))$, becomes zero. In order to solve this problem, we need to do

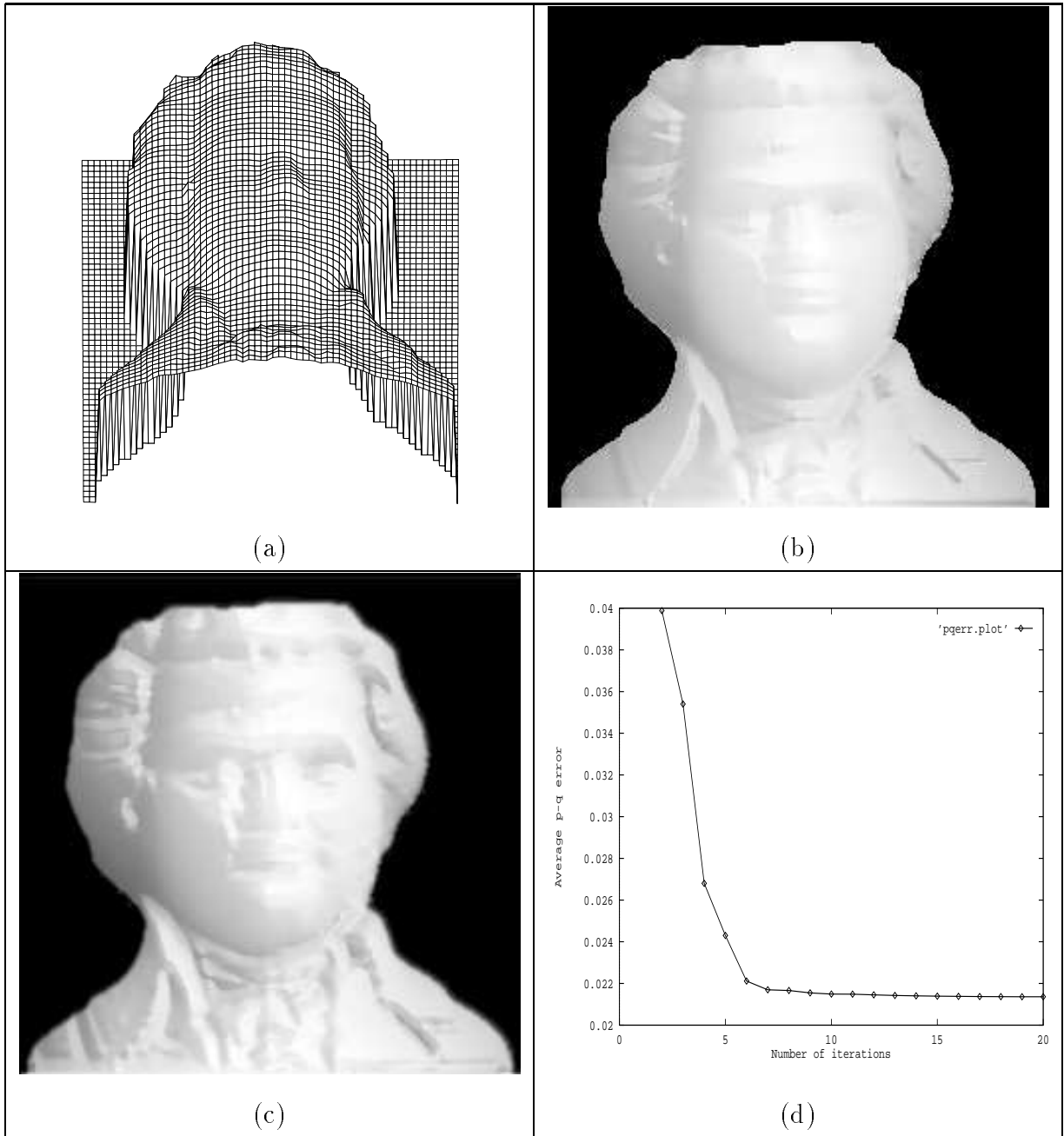


Figure 3: The results for Mozart Image. (a) A 3-D plot of the actual depth map obtained with a laser range finder from USC. (b) The reconstructed gray level image using the actual depth map and light source at (0.01,0.01). (c) The reconstructed gray level image using the estimated depth map and light source at (0.01,0.01). (d) The error-iteration curve.

some modification. Let us rewrite the Equation (9) as follow:

$$Z^n(x, y) = Z^{n-1}(x, y) + K^n(-f(Z^{n-1}(x, y))),$$

where K^n needs to satisfy three constraints. First, K^n is approximately equal to the inverse of $\frac{df}{dZ(x,y)}(Z^{n-1}(x, y))$. Second, K^n equals zero when $\frac{df}{dZ(x,y)}(Z^{n-1}(x, y))$ approaches zero. And third, K^n becomes zero when $Z^n(x, y)$ approaches to true $Z(x, y)$. Now, we define K^n as follow:

$$K^n = \frac{S_{x,y}^n M_{x,y}}{W_{x,y} + S_{x,y}^n M_{x,y}^2},$$

where

$$\begin{aligned} M_{x,y} &= \frac{df}{dZ(x,y)}(Z^{n-1}(x, y)), \\ S_{x,y}^n &= E[(Z^n(x, y) - Z(x, y))^2], \end{aligned}$$

E is the expectation operator, and $W_{x,y}$ is small, but non-zero. Since $W_{x,y}$ is a small value, K^n is approximately equal to $\frac{1}{M_{x,y}}$, which is the inverse of $\frac{df}{dZ(x,y)}(Z^{n-1}(x, y))$. When $M_{x,y}$ approaches to zero, K^n becomes zero. When $Z^n(x, y)$ approaches to true $Z(x, y)$, $S_{x,y}^n$ (the expected value of $(Z^n(x, y) - Z(x, y))^2$) will become zero. Therefore, K^n will be zero. We can see clearly that the definition of K^n satisfies all three constraints.

Many people in the vision community will recognize this as an example of Kalman filtering, which has been applied to many problems in the lower level vision. This can be considered to be Extended Kalman filtering because, in general, the equations for both the state vector and a measurement vector in Kalman filter are nonlinear. However, if good estimates of these vectors are available, a linear approximation in a small neighborhood can be considered. That is precisely what is being done here. In the Kalman filtering terminology, K is known as the Kalman gain, and $W_{x,y}$ and $S_{x,y}$ are the standard deviations associated with the input and the state variables, respectively.

The complete C code for our algorithm with sample images are available by anonymous ftp from `eustis@cs.ucf.edu` under the directory `/pub/shading`. The main part of the program including the iterative loop is only 25 lines. The program runs very efficiently, since all the computations are purely local. For instance, it takes about 0.2 seconds CPU time per iteration on the SUN SPARCstation-1 for a 128×128 image.

6 Experiments

6.1 Results for Lambertian Surfaces

We have applied our algorithm on several real images, and have obtained quite encouraging results. The results are shown in Figures 4–7. In all of these Figures the depth map is shown after two iterations. In these experiments the direction of the light source was computed by using the Lee and Rosenfeld’s method [4], or else the results for illuminant direction quoted in Chellappa and Zheng [12] were directly used.

The results for the Lenna image are shown in Figure 4. This image has been used as a test case in several papers on image compression and shape from shading. In this example the nose, eyes and lips are recovered quite reasonably, as shown in the 3D plot of the depth map in Figure 4.(b). The surface area around cheeks also appear nice and smooth. Two gray level images generated from the recovered depth map using two different light sources are shown in Figures 4.(c)–(d).

Next, the results for the Mannequin image are shown in Figure 5. In this example the head and the surface area around cheeks are recovered reasonably, as shown in the 3D plot of the depth map in Figure 5.(b). Two gray level images generated from the recovered depth map using two different light sources are shown in Figures 5.(c)–(d).

The results of the Yowman image are shown in Figure 6. This is a line drawing of a famous underground cartoon character named Zippy. This image was taken from Pentland’s [8] paper using a standard camcorder, and was then digitized. The recovered 3D surface shown in Figure 6.(b) is amazingly good. The ears, nose, eyes and lips are very clear. These results appear to be slightly better than the results shown by Pentland on the same image. Most parts of Pentland’s 3D plot appear almost flat. Two gray level images generated from the recovered depth map using two different light sources are shown in Figures 6.(c)–(d), which appear very similar to the original gray level image shown in Figure 6.(a).

Finally, the results for the Part image are shown in Figure 7. This is the image of an automobile part. The recovered 3D depth map in Figure 7.(b) clearly shows various surfaces. The round surface in the center appears at a higher depth than the four surface areas shown outside the center.

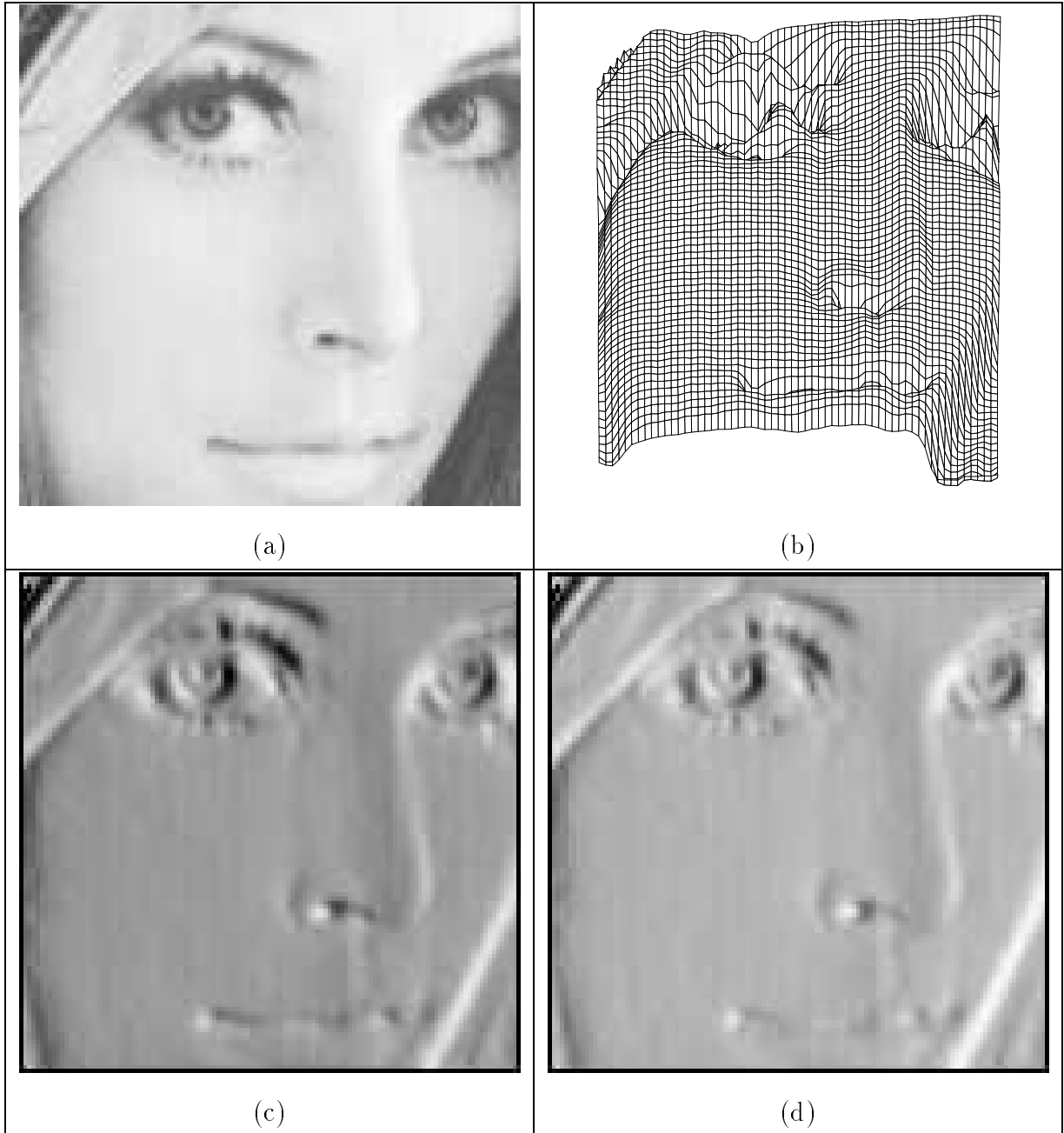


Figure 4: The results for Lenna Image. (a) The input image. The light source parameters estimated by Zheng & Chellappa's method are : $slant = 52.46^\circ$, $tilt = 11.73^\circ$. (b) A 3-D plot of the depth map computed by our algorithm. (c) A reconstructed gray level image using depth map in (b) and constant $albedo = 255$ with the estimated light source direction ($slant = 52.46^\circ$, $tilt = 11.73^\circ$). (d) A reconstructed gray level image using depth map in (b) and constant $albedo = 255$ with the light source direction ($slant = 45^\circ$, $tilt = 0^\circ$).

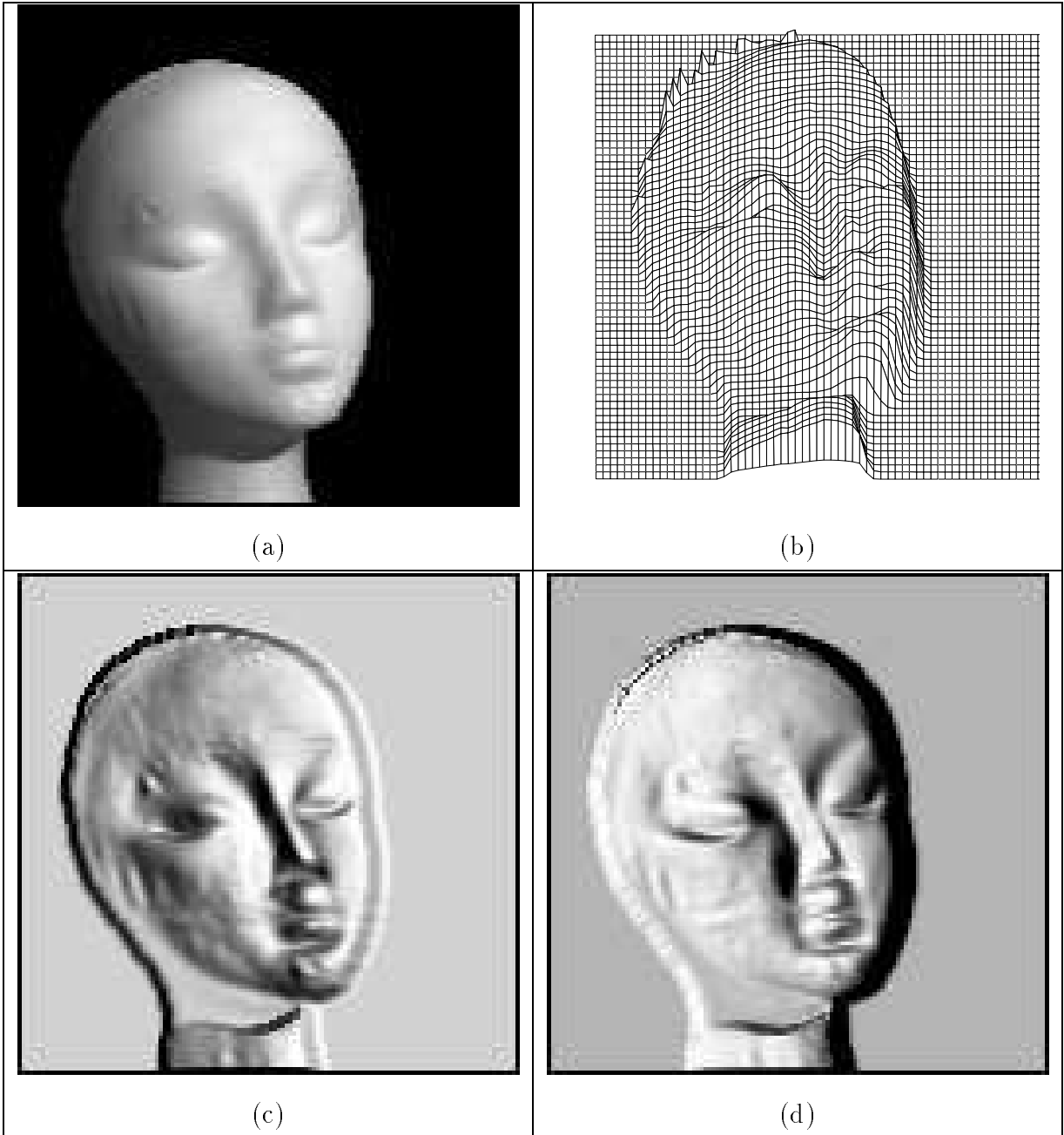


Figure 5: The results for Mannequin Image. (a) The input image. The light source parameters estimated by Lee & Rosenfeld's method are : $slant = 42.2^\circ$, $tilt = 14.4^\circ$. (b) A 3-D plot of the depth map computed by our algorithm. (c) A reconstructed gray level image using depth map in (b) and constant $albedo = 255$ with the estimated light source direction ($slant = 42.2^\circ$, $tilt = 14.4^\circ$). (d) A reconstructed gray level image using depth in (b) and constant $albedo = 255$ with the light source direction ($slant = 135^\circ$, $tilt = 0^\circ$).

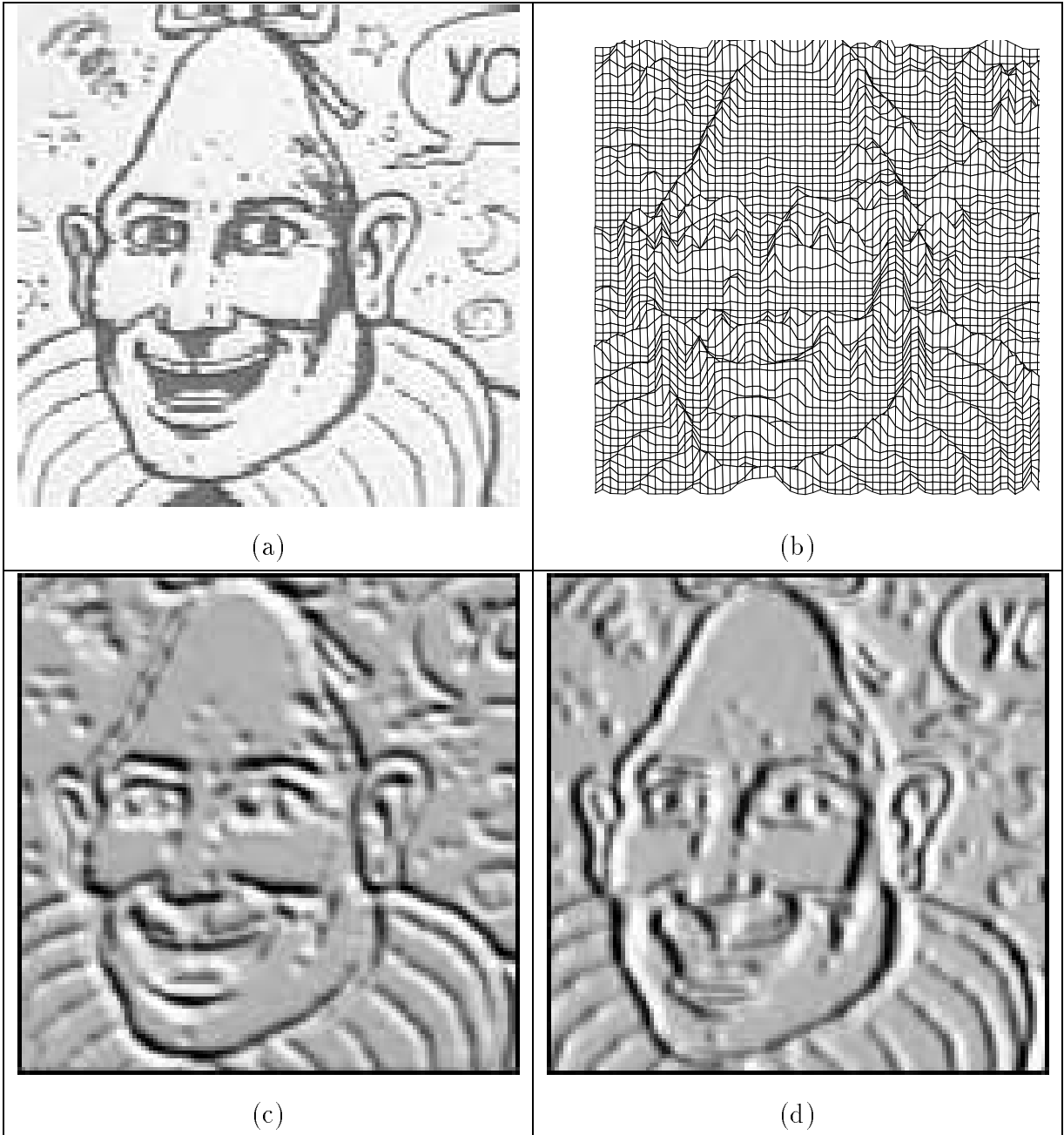


Figure 6: The results for Yowman Image. (a) The input image. The light source parameters estimated by Lee & Rosenfeld's method are : $slant = -45.75^\circ$, $tilt = 62.14^\circ$. (b) A 3-D plot of the depth map computed by our algorithm. (c) A reconstructed gray level image using depth map in (b) and constant $albedo = 255$ with the estimated light source direction ($slant = -45.75^\circ$, $tilt = 62.14^\circ$). (d) A reconstructed gray level image using depth map in (b) and constant $albedo = 255$ with the light source direction ($slant = 45^\circ$, $tilt = 0^\circ$).

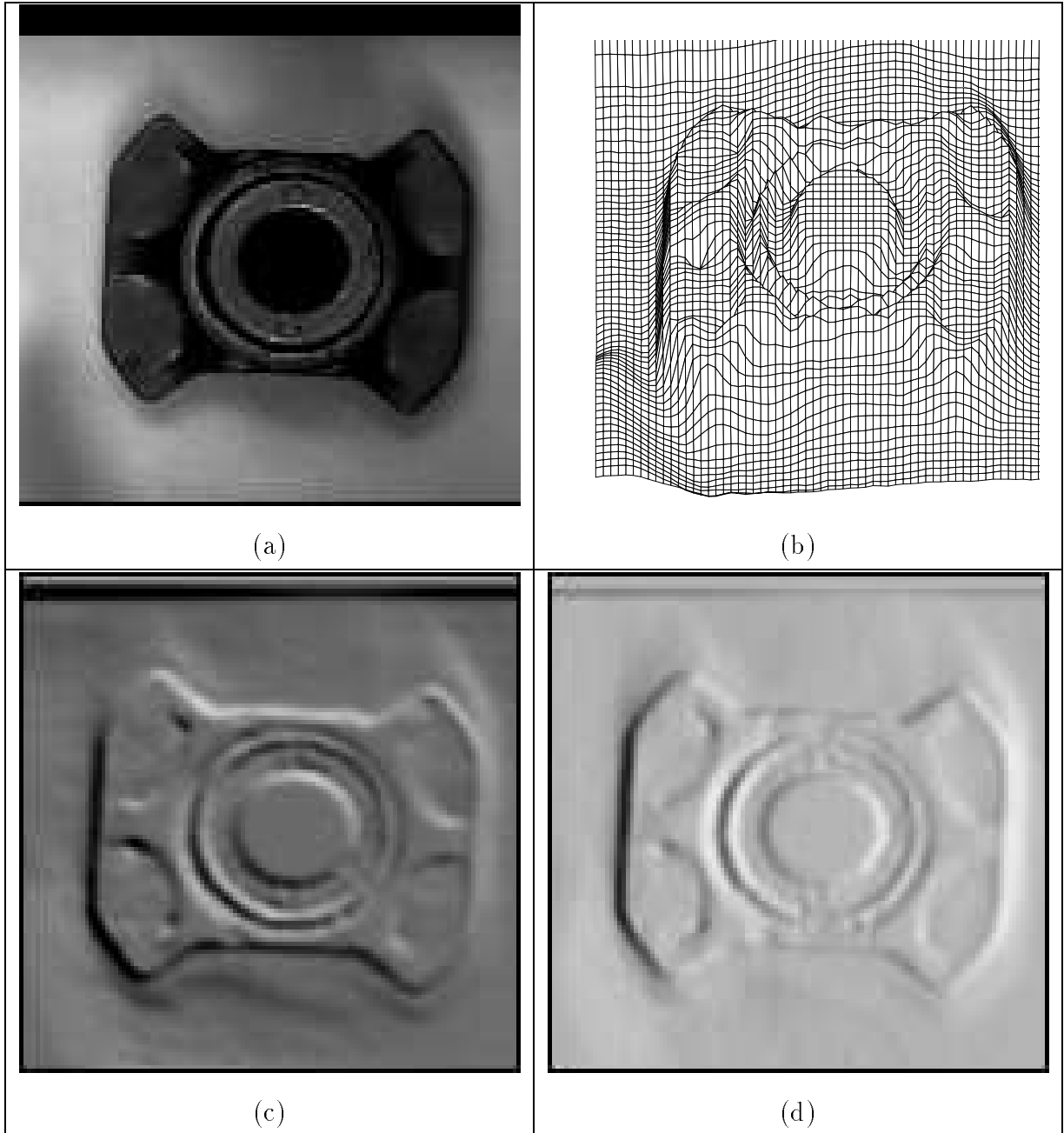


Figure 7: The results for Part Image. (a) The input image. The light source parameters estimated by Lee & Rosenfeld's method are : $slant = 65.28^\circ$, $tilt = 237.94^\circ$. (b) A 3-D plot of the depth map computed by our algorithm. (c) A reconstructed gray level image using depth map in (b) and constant $albedo = 255$ with the estimated light source direction ($slant = 65.28^\circ$, $tilt = 237.94^\circ$). (d) A reconstructed gray level image using depth map in (b) and constant $albedo = 255$ with the light source direction ($slant = 45^\circ$, $tilt = 0^\circ$).

6.2 Result for Specular Surfaces

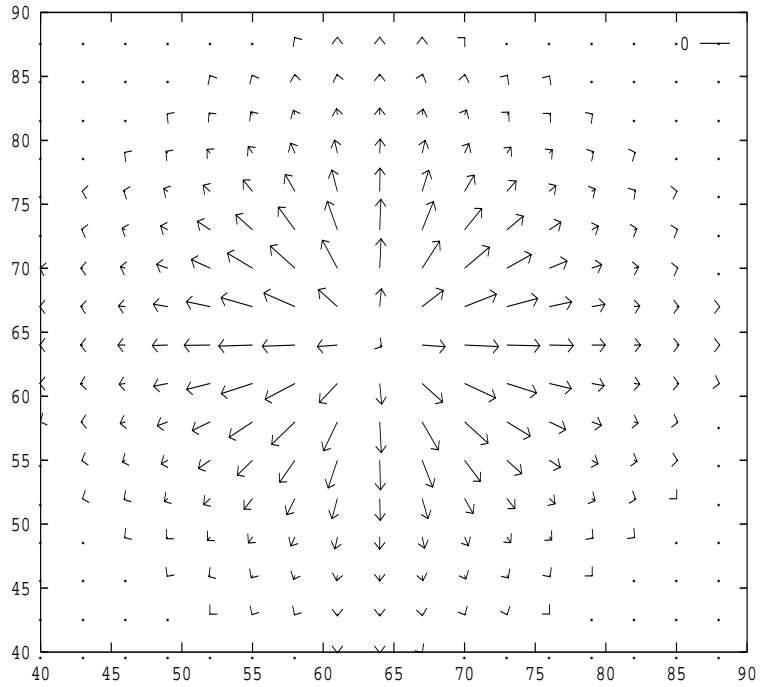
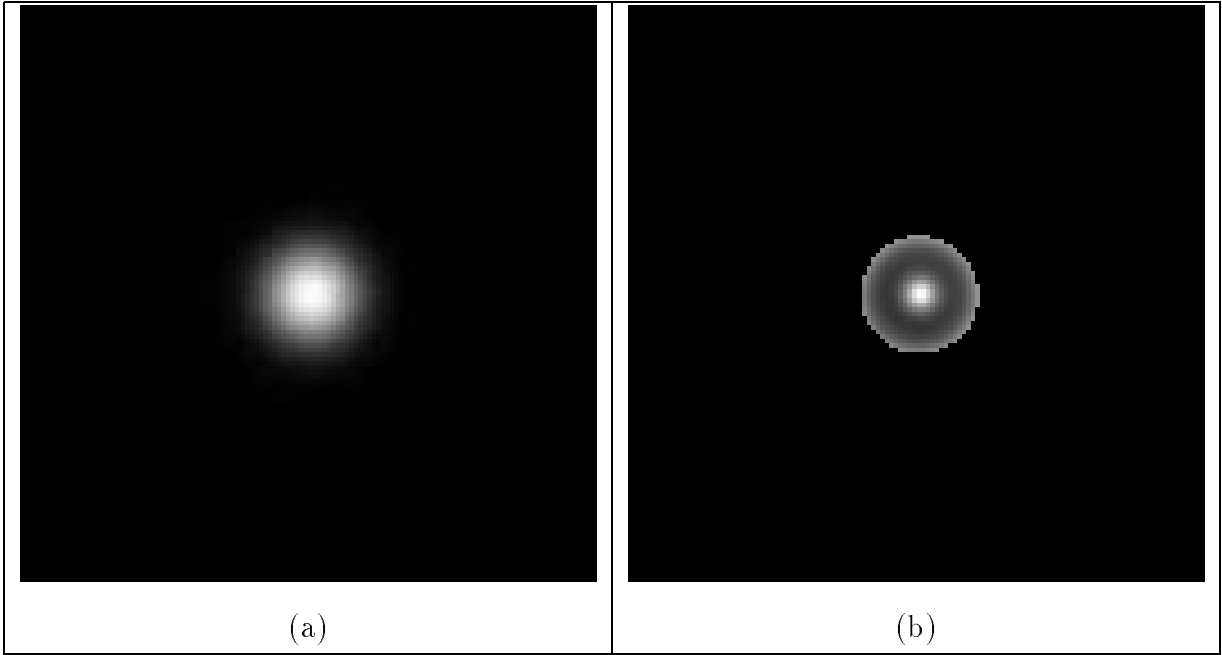
The results for the synthetic specular sphere image are shown in Figure 8. The input image, Figure 8.(a), was generated based on Healey and Binford's reflectance model. The reconstructed gray level image generated from the recovered depth map using the same light source and reflectance model is shown in Figure 8.(b). The scaled needle map of the center area of the image is shown in Figure 8.(c). The needle map clearly shows the shape of a sphere.

The results for the cylinder image are shown in Figure 9. The input image, Figure 9.(a), was taken by a camcorder in our lab. Since we do not have an exact point light source, the image has one wide bright strip instead of a thin line. However, the reconstructed gray level image, Figure 9.(b), looks very similar to the original gray level image.

The results for a tomato image are shown in Figure 10. The input image, Figure 10.(a), was obtained from Carnegie Mellon University. The scaled needle map of the center area of the image is shown in Figure 10.(b).

7 Parallel Implementation

Since our algorithm is purely local, it is very suitable for the parallel implementation. Currently, the algorithm takes .2 seconds for a 128×128 image on the Sun Sparcstation-1. The algorithm can be made real-time by using a parallel machine. We have experimented with two preliminary versions of this algorithm on the BBN GP1000 parallel machine. In the first version, the input image was treated as common memory and was shared by all processors. In the second version, each processor was supplied with a copy of the input image in order to reduce the memory contention. For the first version the single processor of GP1000 took about 17 seconds, due to slow processors in the BBN machine. However, the speedup of 34 with 59 processors was achieved. With the second version, the single processor took about 6 seconds, and a speedup of 29 was obtained with the 32 processors. Figure 11 shows the time-processor curve for the sphere image using the second version of our parallel algorithm. The architecture of GP1000 (MIMD machine) is not very suitable for our problem, and in general it is not appropriate for problems involving image and matrix data structures. In the GP1000 a whole row of an image or matrix is assigned to one process which results in



(c)

Figure 8: The results for a synthetic specular sphere Image. (a) The input image. The light source direction is $(0.01, 0.01, 1)$. (b) A reconstructed gray level image using the estimated depth map with same light source direction. (c) The needle map.

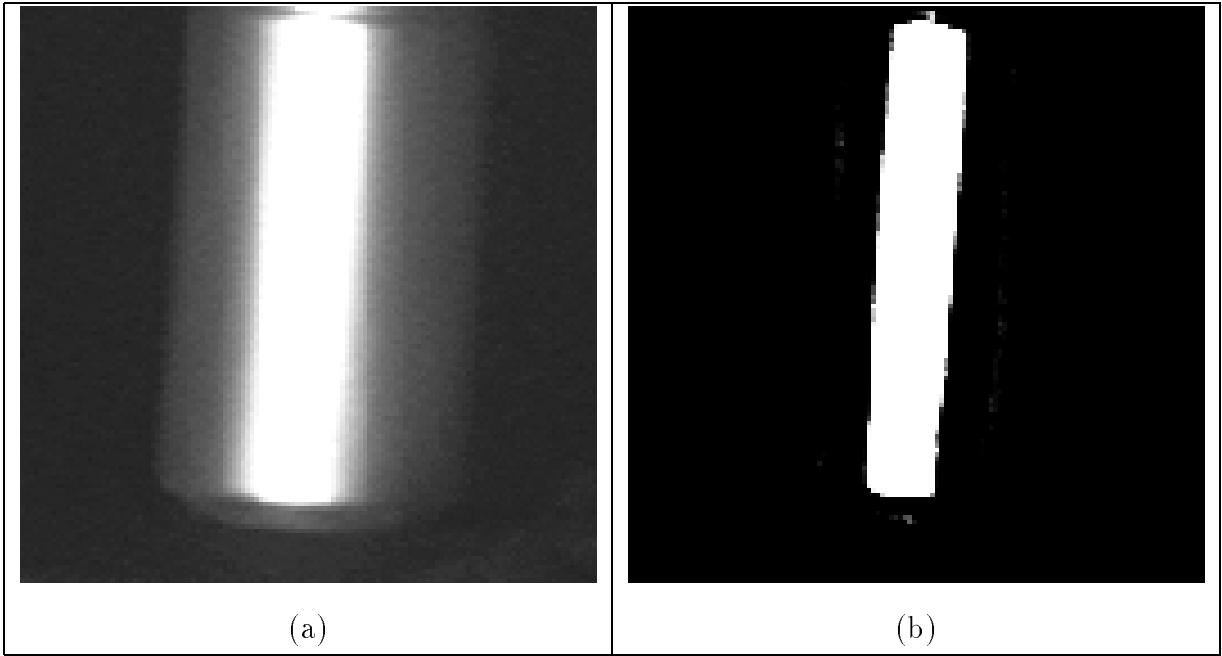


Figure 9: The results for a specular cylinder Image. (a) The input image. The light source direction is approximate $(0,0,1)$. (b) A reconstructed gray level image using the estimated depth map with same light source direction.

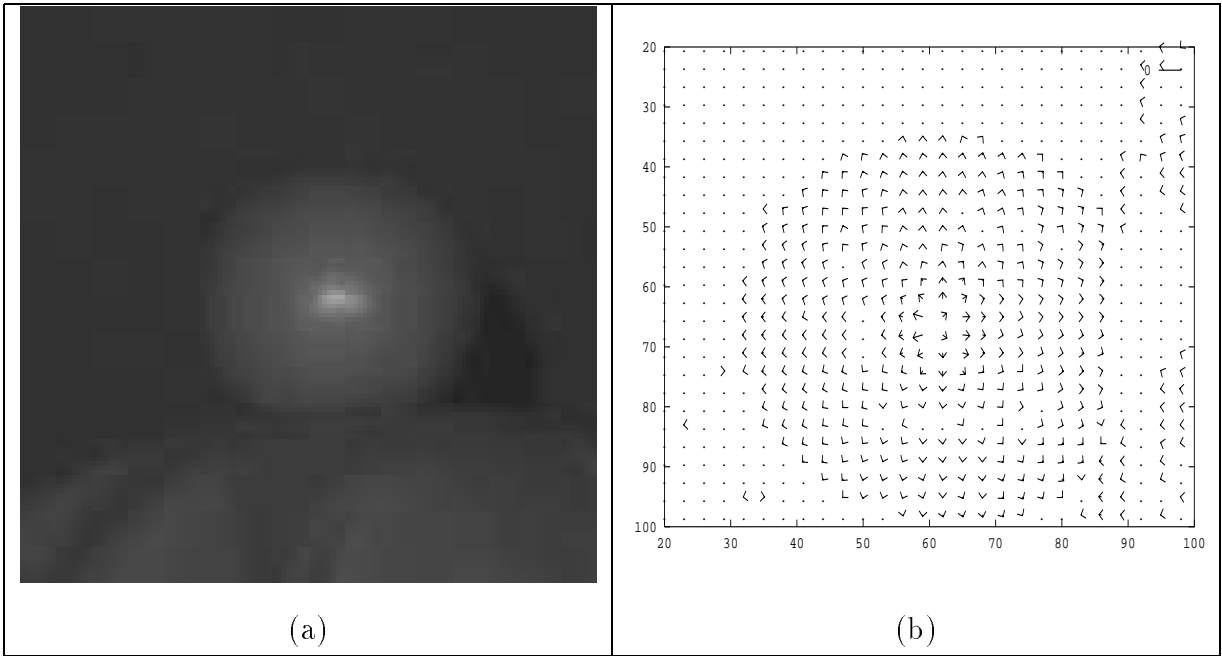


Figure 10: The results for a specular tomato Image. (a) The input image obtained from CMU. The light source direction is $(-0.059,-0.039,0.997)$. (b) The needle map of the center area.

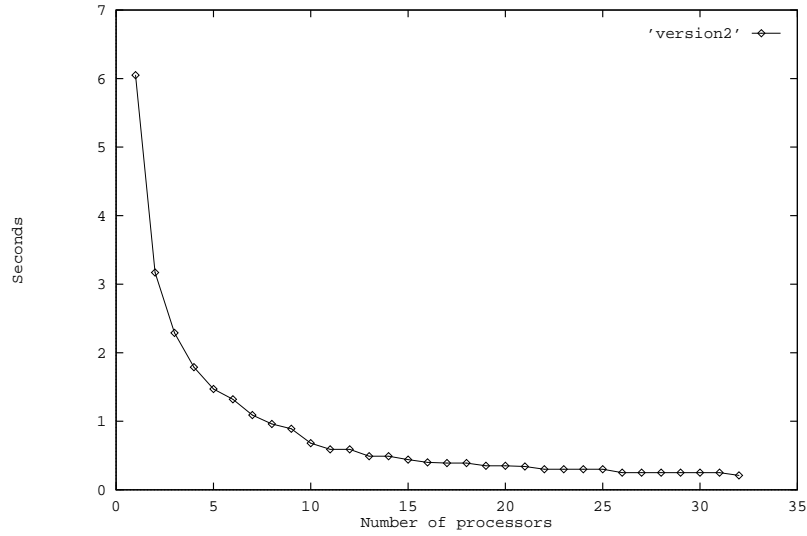


Figure 11: The time-processor curve for the sphere image

memory contention. Note that due to the local nature of our algorithm, a SIMD machine will be more suitable for parallel implementation. The MasPar, a SIMD parallel machine, has some features to facilitate parallelization of algorithms for solving our problem. First, the MasPar machine has a 2-D PE array of 64×64 . Second, each processor has its own distributed local memory. Third, each processor can easily communicate with its neighbors. Therefore, an image can be naturally mapped to this PE array, in which each processor performs the same instructions simultaneously on one or more pixels. We will work on the implementation of our shape from shading algorithm on the MasPar in the future. We expect to achieve a speedup of close to 100% for a 64×64 image with 4,096 processors.

8 Conclusions

The recovery of surface shape from a single shaded image is a very important problem in Computer Vision. We presented a very simple and fast algorithm for computing the depth map from a single monocular image. Our approach uses a linear approximation of reflectance in Z . We first employ a discrete approximation of p and q , and then compute the Taylor series of reflectance up to the first order term. This results in a simple iterative scheme for computing the depth map from an intensity image when the light source direction is known. The algorithm works quite well and is very easy to implement. The results on several real

scenes are presented to demonstrate our method.

Acknowledgements: The authors are thankful to Professor Ikuechi of Carnegie Mellon University for providing the *tomato* image, and Dr. Stein of University of Southern California for providing the *Mozart* image.

References

- [1] M. Bichsel and A. Pentland. A Simple Algorithm for Shape from Shading. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 459–465, June 1992.
- [2] G. Healey and T.O. Binford. Local shape from specularity. *Computer Vision, Graphics, Image Processing*, 42:62–86, 1988.
- [3] B.K.P. Horn. Height and gradient from shading. *International Journal of Computer Vision*, 5:37–75, 1990.
- [4] C.H. Lee and A. Rosenfeld. Improved methods of estimating shape from shading using the light source coordinate system. *AI*, 26:125–143, 1985.
- [5] K.M. Lee and C.-C. J. Kuo. Shape from shading with a linear triangular element surface model. Tech. Rep. 172, USC-SIPI, 1991. Also to appear in *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [6] J. Oliensis. Shape from Shading as a Partically Well-Constrained Problem. *Computer Vision, Graphics, Image Processing: Image Underatanding*, Vol. 54, No. 2, pages 163-183, September, 1991.
- [7] A. Pentland. Finding the illuminant direction. *Journal of Optical Society of America*, 72(4):448–455, 1982.
- [8] A. Pentland. Shape information from shading: a theory about human perception. In *Second International Conference on Computer Vision (Tampa, FL, December 5–8, 1988)*, pages 404–413, Washington, DC, 1988. Computer Society Press.
- [9] A. Pentland. Photometric motion. In *Proc. International Conference on Computer Vision*, pages 178–187, December 1990.
- [10] B. T. Phong. Illumination for computer generated pictures. *Comm. of ACM*, 18:311–317, 1975.
- [11] K. E. Torrance and E. M. Sparrow. Theory for off-specular reflection from roughened surfaces. *J. Of Optical Society Of America*, 57:1105–1114, 1967.
- [12] Q. Zheng and R. Chellapa. Estimation of Illuminant Direction, Albedo, and Shape from Shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 7, pages 680–702, July 1991.