from IEEE Pattern Analysis and Machine Intelligence, 19(9):1020-1025, September 1997

Towards Accurate Recovery of Shape from Shading under Diffuse Lighting

A. James Stewart and Michael S. Langer

Abstract—

A new surface radiance model for diffuse lighting is presented which incorporates shadows, interreflections, and surface orientation. An algorithm is presented that uses this model to compute shape–from–shading under diffuse lighting. The algorithm is tested on both synthetic and real images, and is found to perform more accurately than the only previous algorithm for this problem.

Index Terms: shape–from–shading, diffuse lighting, interreflections, shadows, visual events, horizon, skyline

I. Introduction

The classical formulation of the shape–from–shading problem has been to assume that surface radiance is determined entirely by the surface orientation relative to a point light source at infinity [5], [7]. Shadows and interreflections are usually ignored. When the light source is diffuse, however, this model is not applicable [9], [10]. To see this, consider the following two examples: The first is a scene consisting of a convex Lambertian object resting on a ground plane, illuminated from above by a uniform hemispheric source. It is argued by Horn and Sjoberg [6] that, because of self–shadowing, the direct component of radiance on the object is determined by the surface normal. While this is true for the object, it is *untrue* for the ground plane. The direct component of radiance on the ground plane varies continuously because of cast shadows, without any change in surface normal. The second example is a spherical concavity excavated from a ground plane. A surprising result is that when shadowing, surface orientation, and interreflection effects are all modelled, surface radiance is *constant* within the concavity [11]. In particular, the interior of the concavity has the same local appearance under diffuse lighting as would a plane under point-source-at-infinity lighting. It is clear from these examples that, under diffuse lighting, surface normal variations are neither necessary nor sufficient for radiance variations.

II. Problem Formulation

We address the following scenario. Suppose a Lambertian surface with albedo, ρ, is illuminated by a uniform hemispheric source of radiance, R_{src}. We assume that the source does not itself reflect light, nor is light attenuated between the source and the surface. The surface radiance, $R(\mathbf{x})$, is then determined by the radiosity equation [13], [2]:

$$R(\mathbf{x}) = \frac{\rho}{\pi} \int_{\mathcal{V}(\mathbf{x})} R_{src}\ \mathbf{N}(\mathbf{x}) \cdot \mathbf{u}\ d\Omega + \frac{\rho}{\pi} \int_{\mathcal{H}(\mathbf{x}) \setminus \mathcal{V}(\mathbf{x})} R(\Pi(\mathbf{x}, \mathbf{u}))\ \mathbf{N}(\mathbf{x}) \cdot \mathbf{u}\ d\Omega \qquad (1)$$

where: $\mathbf{x}$ is a surface point; $\mathbf{N}(\mathbf{x})$ is the surface normal; $\mathcal{H}(\mathbf{x}) = \{\mathbf{u} : \mathbf{N}(\mathbf{x}) \cdot \mathbf{u} > 0\}$ is the hemisphere of outgoing unit vectors; $\mathcal{V}(\mathbf{x})$ is the set of unit directions in which the diffuse source is visible from $\mathbf{x}$; $d\Omega$ is an infinitesimal solid angle; $\Pi(\mathbf{x}, \mathbf{u})$ is the surface point visible from $\mathbf{x}$ in direction $\mathbf{u}$ (Π denotes "projection").

A. Interreflection Model

The first contribution of this paper is to approximate Equation (1) by removing the dependence on $\Pi(\mathbf{x}, \mathbf{u})$. To motivate intuitively the approximation, consider a sphere resting on a plane. Observe that the lower half of the sphere tends to be dark for two reasons: because of self–shadowing and because it is illuminated by darker points immediately below it on the ground plane. Similarly, points on the upper half of the sphere tend to be bright both because little self–shadowing occurs and because the interreflections that do occur are from distant unshadowed points on the ground plane. A similar observation holds for a smooth depth map. From a hilltop, one mostly sees other hilltops, while within a valley, one mostly sees the valley.

These observations suggest the following heuristic to simplify Equation (1): *Under diffuse lighting and constant albedo, a point on a surface tends to be illuminated by other points having similar radiance.* This heuristic can be formalized as follows. Approximate the right hand side of Equation (1) by replacing the incoming radiance, $R(\Pi(\mathbf{x}, \mathbf{u}))$, from other surfaces by the outgoing radiance, $R(\mathbf{x})$, from $\mathbf{x}$ itself. An algebraic manipulation yields the **interreflection model:**

$$R_1(\mathbf{x}) \equiv \frac{\rho\ R_{src}\ \frac{1}{\pi} \int_{\mathcal{V}(\mathbf{x})} \mathbf{N}(\mathbf{x}) \cdot \mathbf{u}\ d\Omega}{1 - \rho\ (1 - \frac{1}{\pi} \int_{\mathcal{V}(\mathbf{x})} \mathbf{N}(\mathbf{x}) \cdot \mathbf{u}\ d\Omega)}\ . \qquad (2)$$

The model is exact for the case of the spherical concavity since surface radiance is constant within such a concavity. It is also exact for a surface of arbitrary geometry in the limit of $\rho \to 0$ since the interreflection component vanishes, as well as in the limit of $\rho \to 1$ since $R(\mathbf{x}) \to R_{src}$. For intermediate albedos and more general surface shapes, we empirically test the interreflection model by rendering depth maps with a variant of

©1997 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

A. James Stewart, Department of Computer Science, University of Toronto, 10 Kings College Road, Toronto, Ontario, Canada, M5S 1A4. phone: (416) 978-5359. email: jstewart@cs.toronto.edu

Michael S. Langer, Computer Science Research Division, NEC Research Institute, Inc., 4 Independence Way, Princeton, NJ 18540, USA. email: langer@research.nj.nec.com

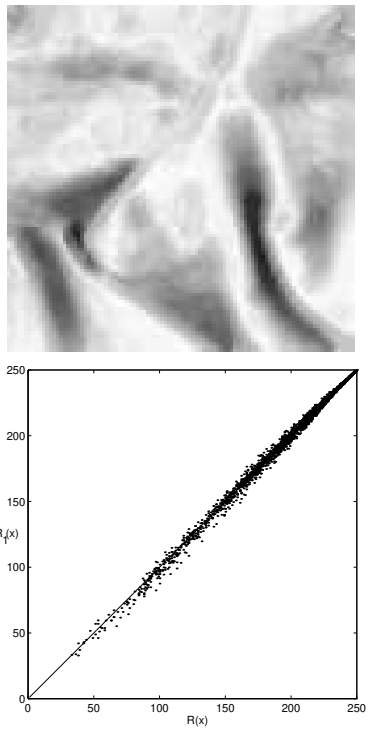

Fig. 1. A synthetic 100 × 100 smooth depth map of a drapery was rendered using the radiosity equation $R(\mathbf{x})$ with $\rho = 0.5$. The scatter plot compares $R_1(\mathbf{x})$ to $R(\mathbf{x})$.

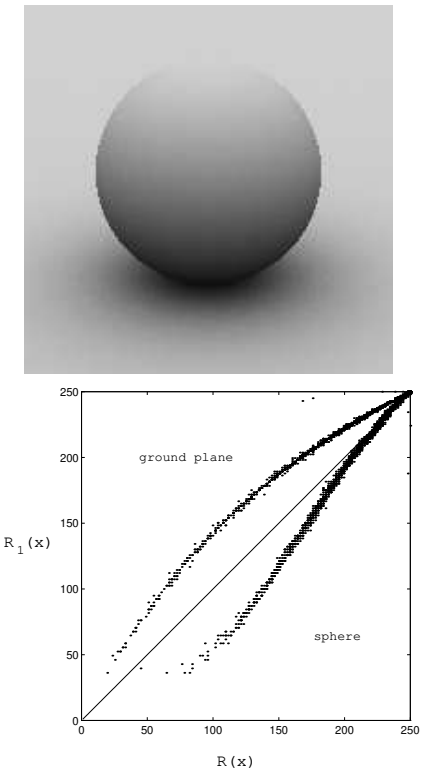

Fig. 2. A rendered image of an sphere resting on a ground plane, and a scatter plot comparing $R_1(\mathbf{x})$ to $R(\mathbf{x})$ in this scene.

the computer graphics radiosity algorithm [2]. Figure 1 shows a rendered image of a smooth depth map with $\rho = 0.5$, along with a scatter plot of $R(\mathbf{x})$ versus $R_1(\mathbf{x})$. Note the high correlation between the two. Similar plots were observed for the other smooth depth maps tested.

The R_1 model breaks down is some situations. For example, Figure 2 shows a scatter plot of $R_1(\mathbf{x})$ versus $R(\mathbf{x})$ for a scene consisting of an sphere sitting on a ground plane. (The scene was rendered using the RADIANCE lighting system [15].) A scatter plot is also shown. Two clusters occur, corresponding to points on the ground plane and points on the sphere. The R_1 model overestimates the radiance of the ground plane but underestimates the radiance of the sphere.

B. Image Formation Model

We assume that images are formed as follows. Image intensities are linearly related to surface radiance (we ignore sensor calibration issues). The surface seen in the image is Lambertian with known albedo, ρ. The image is formed under orthographic projection. The depth map $z(x, y)$ is a continuous, single–valued function of (x, y). The source is a uniform hemisphere centered about the line of sight with its equator at depth zero. Surfaces outside the image domain do not cast shadows on surfaces within the image domain. From these six assumptions, it follows that the highest image intensity, I_{max}, corresponds to a surface point that sees the entire hemispheric source [10]. Moreover, from Equation (2), we have the following image formation model:

$$I(x,y) = \frac{I_{max}\ \frac{1}{\pi}\ \int_{\mathcal{V}(\mathbf{x})}\ \mathbf{N}(\mathbf{x})\cdot\mathbf{u}\ d\Omega}{1\ -\ \rho\ (\ 1-\frac{1}{\pi}\ \int_{\mathcal{V}(\mathbf{x})}\ \mathbf{N}(\mathbf{x})\cdot\mathbf{u}\ d\Omega\ )} \qquad (3)$$

The problem we address is how to compute a depth map $z(x, y)$ which is consistent with a given input image, $I(x, y)$, and with the image formation model.

III. New Algorithm

The only previous algorithm for solving the shape–from–shading problem under diffuse lighting is due to Langer and Zucker (LZ). The reader is referred to their papers for details [9], [10]. The algorithm we introduce is similar in spirit to the LZ algorithm. Each pixel is associated with a **node** $\mathcal{N}(x, y)$ in space, such that the depth value $z(x, y)$ of that node increases monotonically from zero as the algorithm progresses. Over the course of the algorithm, each node descends until it reaches the surface depth z that satisfies Equation (3). The key improvement of the new algorithm over the LZ algorithm is that the new algorithm accounts for surface normal effects. It does so by using a more accurate, continuous representation of the depth variable (whereas the LZ algorithm uses discrete depths with a fixed interval), a continuous representation of the visible sky variable (whereas the LZ algorithm samples a hemicube), and a more sophistic-

ated set of routines for computing the depths and visible sky.

A. Visual Events

The visible sky, $\mathcal{V}(\mathbf{x})$, of a node at position $\mathbf{x}$ is represented by its one–dimensional boundary curve, which we call the **skyline**. From the viewing assumptions that led to Equation (3), it follows that a skyline is a single–valued function of the azimuth angle.

The algorithm divides the skyline of a node by azimuth angle into a number of equal–size sectors. There are typically 16 sectors, each spanning $\frac{\pi}{8}$ radians. The algorithm approximates the skyline in each sector with a constant elevation angle, which is the highest elevation angle of the surface nodes visible in that sector. We implicitly assume that each surface node spans the entire sector. Accuracy can be traded for speed by increasing the number of sectors. See Figure 3.

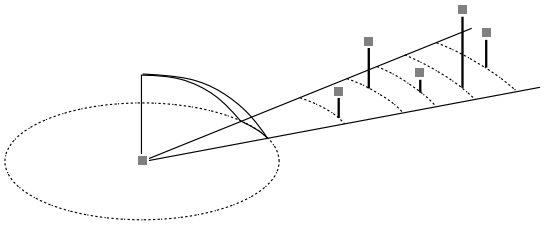

Fig. 3. The skyline around node $\mathcal{N}$ is divided into sectors. The skyline within each sector is approximated by a constant elevation angle, which is the highest elevation angle of all surface nodes (grey in the figure) in that sector.

As the depth of N increases, we must continually maintain the skyline so as to determine, at any depth, whether Equation (2) is satisfied. Rewriting Equation (2) to isolate the visible sky, $\mathcal{V}(\mathbf{x})$, we get

$$\int_{\mathcal{V}(x,y,z)} \mathbf{N}(x,y,z) \cdot \mathbf{u} \; d\Omega \;=\; \frac{\pi\ (1-\rho)\ I(x,y)}{I_{max} - \rho I(x,y)}. \quad (4)$$

Consider one sector of the skyline of a node $\mathcal{N}$. As the depth of $\mathcal{N}$ increases, closer surface nodes in the sector obscure more distant surface nodes, since surface nodes remain fixed while $\mathcal{N}$ descends and since each surface node is assumed to span the whole sector. When the surface node of highest elevation within a sector becomes obscured, a **visual event** occurs and the skyline of $\mathcal{N}$ undergoes a qualitative change and must be updated (see Figure 4). Visual events were introduced by Kœnderink and van Doorn [8] and developed further by Plantinga and Dyer [12] and Gigus and Malik [4].

To facilitate this update, each sector of $\mathcal{N}$ stores the surface nodes that can potentially appear on the skyline, sorted by increasing horizontal distance. The nodes in this list form a convex chain, as shown in Figure 5. Clearly, no node below the convex chain can appear on the skyline since it will be obscured by a closer node. Also, the most distant node in a convex chain has maximum elevation and appears on the skyline of the sector. As the depth of $\mathcal{N}$ increases, each node on the convex chain appears at the skyline in turn, in order of decreasing horizontal distance to $\mathcal{N}$.

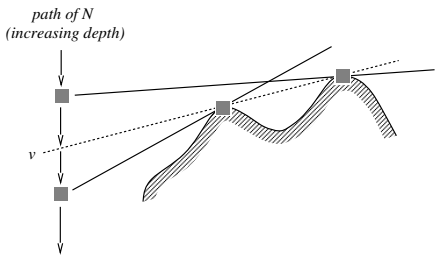

Fig. 4. As the depth of a node $\mathcal{N}$ increases, the surface defining the skyline of $\mathcal{N}$ changes. A visual event occurs at the point of change (v in the figure). The higher node appears on the skyline when $\mathcal{N}$ is above v and the lower node appears when $\mathcal{N}$ is below.

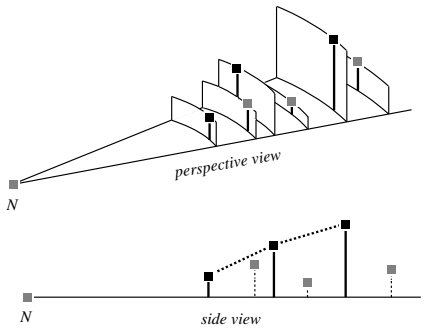

Fig. 5. Each sector of $\mathcal{N}$ stores a convex chain of nodes that can potentially appear on the skyline. Nodes below the chain cannot appear on the skyline and are not stored, while nodes on the chain appear on the skyline in order of decreasing horizontal distance as $\mathcal{N}$ descends. In the figure, the black nodes are on the convex chain.

If a sector contains at least two surface nodes on its convex chain, the highest visual event is computed in a straightforward manner: Let z_1 be the depth of the most distant node on the chain. Let d_1 be its horizontal distance from $\mathcal{N}$. Define z_2 and d_2 similarly for the next–most–distant node on the chain. Then the visual event occurs at depth $z_2 + d_2\ (z_1 - z_2)/(d_2 - d_1)$. Each sector stores the depth of its highest visual event.

B. Surface Events and Normal Estimation

A **surface event** occurs when a node $\mathcal{N}(x,y)$ reaches a depth z that satisfies Equation (4). In this section we discuss how to determine z, given the skyline around $\mathcal{N}(x,y)$. The right side of Equation (4) is computed once for each (x,y). Assume, for the moment, that we can compute the integral on the left side for any particular value of z. Then the solution of the equation can be determined by binary search, as follows. Let z_{min} be a lower bound on z and let z_{max} be an upper bound. That is, the integral is greater than (less than) the right side, when evaluated at z_{min} (z_{max}). Let $z_{mid} = (z_{min} + z_{max})/2$. Evaluate the integral in Equation (4) at z_{mid}. If

this value is greater than the value of the right side assign z_{mid} to z_{min}; otherwise assign z_{mid} to z_{max}. Continue until the difference between z_{min} and z_{max} is below some threshold (we use 10^{-4}).

We next describe how to compute the integral on the left side, given a depth z and a node $\mathcal{N}(x,y)$ for which the skyline is known. First, the surface normal $\mathbf{N}(x,y,z)$ is estimated by considering the nodes in the 8–neighborhood of (x,y). For each *pair* of such nodes whose final depths are already known, we determine the plane that passes through the two nodes and the point (x,y,z). The normals of all such planes are averaged to yield the estimate for $\mathbf{N}(x,y,z)$. If only one adjacent surface node exists, we use the "most upward pointing" normal to the line passing through (x,y,z) and the node (that is, the normal of most negative z component). If no surface nodes exist, we use $(0,0,-1)$ as the normal.

Given the normal, the integral is computed as follows. The skyline is divided into k sectors $0,1,\ldots,k-1$, such that sector i spans the azimuth angle $[\frac{2\pi}{k}i, \frac{2\pi}{k}(i+1))$. Let ϕ_i be the elevation angle, measured from the vertical, of the most distant node $\mathcal{N}_i$ in the convex chain of sector i as seen from the point (x,y,z). Then

$$\begin{aligned}
&\int_{\mathcal{V}(x,y,z)} \mathbf{N}(x,y,z)\cdot \mathbf{u}\, d\Omega \\
&\approx \mathbf{N}(x,y,z)\cdot \sum_{i=0}^{k-1} \int_{\frac{2\pi}{k}i}^{\frac{2\pi}{k}(i+1)} \int_0^{\phi_i} (\sin\phi\cos\theta, \sin\phi\sin\theta, \cos\phi)\sin\phi \; d\phi \; d\theta \\
&= \tfrac{1}{2}\mathbf{N}(x,y,z)\cdot \sum_{i=0}^{k-1} ((\phi_i - \tfrac{\sin 2\phi_i}{2})\Delta\mathrm{SIN}_i, (\phi_i - \tfrac{\sin 2\phi_i}{2})\Delta\mathrm{COS}_i, \tfrac{2\pi}{k}\sin^2\phi_i)
\end{aligned}$$

where $\Delta\mathrm{SIN}_i = \sin\frac{2\pi}{k}(i+1) - \sin\frac{2\pi}{k}i$ and $\Delta\mathrm{COS}_i = \cos\frac{2\pi}{k}i - \cos\frac{2\pi}{k}(i+1)$ are computed once and stored.

C. Priority Queue of Events

The algorithm maintains a priority queue of visual and surface events, where the priority of an event is the depth at which it occurs. For each pixel (x,y) of the image, only the highest event involving the node $\mathcal{N}(x,y)$ is stored in the queue. The algorithm proceeds by removing and processing events from the queue in order of increasing depth. When the queue becomes empty, all nodes will have a depth z that satisfies Equation (4) and the algorithm terminates.

Initially, nodes $\mathcal{N}(x,y)$ corresponding to pixels with maximum intensity are inserted into the queue as surface events of priority zero (recall that these nodes are assumed to have depth zero). As described in Section III-E below, removal of a surface event from the queue may cause more events to be inserted, so the size of the queue varies over time.

D. Processing a Visual Event

Consider a visual event that occurs at depth z_{event} in sector i of node $\mathcal{N}$. When $\mathcal{N}$ falls below this depth, the highest surface node in sector i becomes obscured by a closer surface node and $\mathcal{N}$'s skyline must be updated. The visual event is processed as follows:

1. Delete the most distant surface node in the convex chain of sector i.
2. Calculate and store the depth of the next visual event in sector i. This depth depends on the depths and distances of the two most distant nodes on the convex chain.
3. Determine the sector j of node $\mathcal{N}$ with the highest visual event. This may or may not be sector i. Let z_{vis} be the depth of this event.
4. Compute the integral on the left side of Equation (4) at z_{vis}. If this integral is greater than the right side, insert into the priority queue a new visual event for sector j of node $\mathcal{N}$. Otherwise, a surface event must occur above z_{vis}; in this case, compute the depth of the surface event (as described in Section III-B) and insert into the priority queue a new surface event for $\mathcal{N}$.

E. Processing a Surface Event

A surface event occurs to node $\mathcal{N}$ when Equation (4) is satisfied. Node $\mathcal{N}$ will remain stationary while the remaining **free nodes** (those whose depth is not yet known) will continue to descend. Thus, node $\mathcal{N}$ may appear on the skylines of remaining free nodes. For each remaining free node $\mathcal{N}_{free}$, perform the following:

1. Determine the sector i of $\mathcal{N}_{free}$ in which $\mathcal{N}$ appears.
2. If $\mathcal{N}$ is more distant than the closest surface node on the convex chain of sector i then nothing needs to be done, since $\mathcal{N}$ will always be below the convex chain and so will never appear on the skyline of $\mathcal{N}_{free}$. Otherwise, perform the following steps:
   (a) Add $\mathcal{N}$ to the front of the convex chain of section i.
   (b) Addition of $\mathcal{N}$ may cause the chain to become non–convex. Remove those nodes of the chain that are not vertices in the new convex hull.
   (c) If the second–to–last node on the chain has changed, the highest visual event in the sector will have changed. In this case, remove from the priority queue the event associated with node $\mathcal{N}_{free}$ and compute a new event for that node (by performing on $\mathcal{N}_{free}$ Steps (2) through (4) in Section III-D). Insert the new event into the priority queue.

Related work in computational geometry has been reported by Cole and Sharir [3] and Bern *et al* [1], who answer ray shooting queries from points on vertical flightpaths. However, we are interested only in detecting those points on vertical flightpaths at which the horizon changes. Unlike the situation that they address, we do *not* have the depth map provided ahead of time, and so cannot build a complete data structure beforehand (as they do). Furthermore, while they treat a single vertical flightpath, we are concerned with one flightpath per pixel.

IV. Results

We now compare the performance of the LZ and the new algorithm on several test images, each of size 100×100 pixels. For each image, we show a cross–section of the real depth map and of the depth maps computed by the LZ and the new algorithms.

Figure 6 shows a grey–level image of a hemispherical concavity excavated from the ground (recall the example

in Section I). Note the smooth, accurate depth map that results from the new algorithm, which represents depth as a continuum.

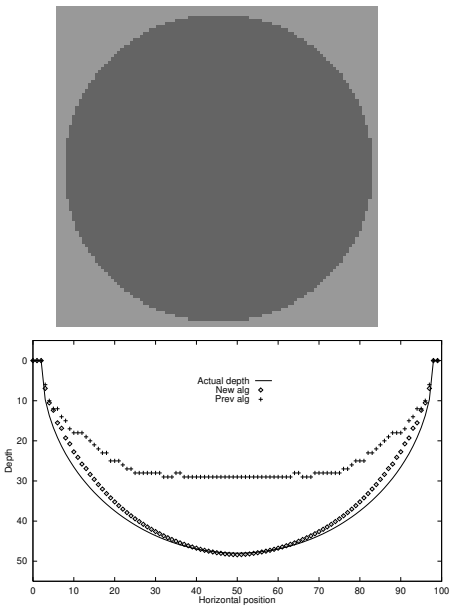

Fig. 6. An analytically computed image of a spherical concavity ($\rho = 0.5$) and a cross section of the depth map on row 50.

Figure 7 shows the results of the algorithm on the smooth depth map of Figure 1. Observe that both algorithms underestimate the depth of central hilltop. This is due to a subtle ill–conditioning property of the shape–from–shading–under–diffuse–lighting problem. Small differences in image intensity of the brightest points in the image can correspond to relatively large differences in depth. For example, over the entire surface, the depths range from 3 to 39 and the intensities range from 32 to 250. Pixel $(x, y) = (15, 4)$, which is a local intensity maximum, has depth 4 and intensity 249, while pixel $(x, y) = (90, 71)$, also a local intensity maximum, has depth 13 and intensity 247. In this case, a 0.9% difference in intensity corresponds to a 25% difference in depth. Such a small intensity difference is typically lost in the image noise.

Observe that both algorithms overestimate the depth at the right boundary of the image. Because there is a local intensity maximum on the right boundary, both algorithms assume the surface is horizontal. (Since both algorithms construct the depth map in order of increasing depth, the darker pixels below the local maximum are not processed until the local maximum is fixed. In particular, in the new algorithm, the deeper pixels cannot contribute to the normal estimation.) From the cross–section, it is clear that the normal should be pointing leftward. Because of this error, the surface receives less light than the algorithms expect, so the algorithms infer a smaller solid angle of the visible source and hence a greater depth.

Finally, we compare the LZ and new algorithms on a

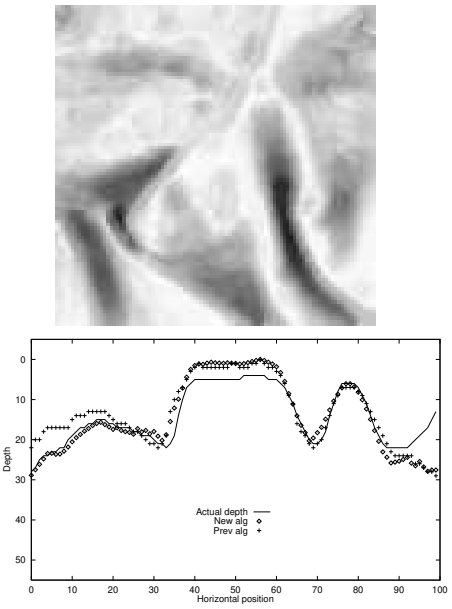

Fig. 7. Fig. 1 (repeated) and a cross section of the depth map on row 50.

real image. Figure 8 shows two eggs resting on a ground plane, viewed from above by a Sony CCD video camera. The eggs and ground were painted grey and matte with albedo 0.7. The scene and camera were surrounded by a large white bedsheet which acted as a hemispheric diffuse light source. Since the distance to the bedsheet (1 meter) was much larger than the size of the scene, the interreflections between the bedsheet and surfaces were negligible.

Calibrated depth maps were obtained directly from the images using the fact that eggs have one degree of rotational symmetry. Pixel noise was reduced to one grey–level out of 255 by averaging over multiple images. A range of camera apertures was used to verify the linearity of the sensor response. Vignetting effects near the image boundary were removed by dividing the image intensities by those of a second image containing only the grey ground. In Figure 8, observe that the new algorithm was able to recover the height of the ground plane more accurately than the LZ algorithm.

Table I summarizes the mean and root–mean–square (RMS) errors for each of the images discussed above. Error is measured in pixel–width units. The true depth maps were between 35 and 50 units deep at their deepest point. The data clearly show that the new algorithm performs better than the LZ algorithm. Further discussion appears in [14].

The new algorithm took between 20 and 36 minutes to process the 100 by 100 images described in this section. It was between 1.5 and 3 times slower than the LZ algorithm, depending upon the image. In the new algorithm, about 15% of the time was spent processing visual events, while surface events took the remaining 85%. Each surface event is expensive because it involves

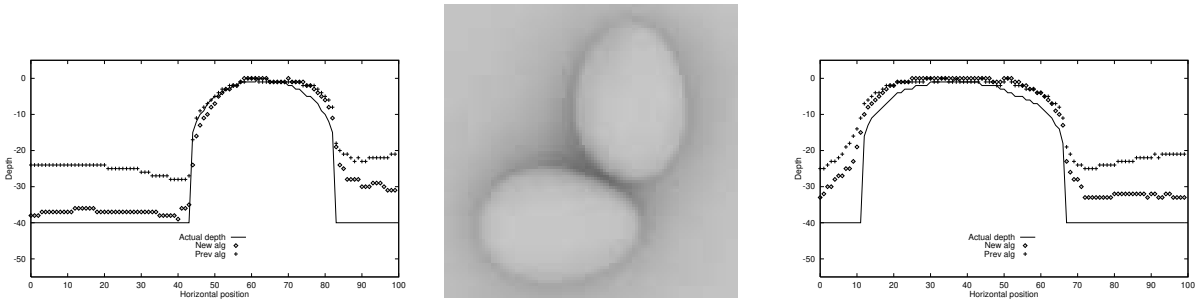

Fig. 8. A real image of two eggs resting on a ground plane, viewed from above. Both eggs and ground were painted grey with $\rho = 0.7$. Horizontal slices through the actual and computed depth maps are shown for rows 36 and 75.

TABLE I
Error Measurements (in pixel-width units)

Scene	Algorithm	Mean Error	RMS Error
Spherical	New	1.5	1.7
Concavity	LZ	7.6	6.1
Smooth	New	-2.5	4.7
	LZ	-0.1	5.5
Two Eggs	New	6.0	6.0
	LZ	12.1	7.9

inserting a new surface node into the skyline of every other node. In all, the algorithm performs about n^2 such insertions for an image of n pixels. Since many of the insertions do not result in a new node on the convex chain of the skyline (Step 2 in Section III-E), an interesting problem is to develop an algorithm which only performs the necessary insertions.

V. Conclusion

The improvements of our algorithm over the LZ algorithm in the drapery and real egg scenes are modest. This suggests that the three types of errors we have discussed — namely, approximations in the image formation model, poor local constraints at the image boundary, and ill–conditioning of the problem itself — provide an ultimate bound on the algorithm's performance. Moreover, we did not consider the case that the albedo was estimated incorrectly or that the hemispheric source was not aligned with the viewing direction. We expect these to introduce further errors in the computed results.

Those warnings aside, we should keep in mind that we have restricted ourselves to shading information alone. We expect that by combining our algorithm with one based on stereo, motion, or other cues, it would be possible to obtain more accurate results than what we have achieved. This remains a subject for future work.

Acknowledgements

James Stewart's work is supported by the Information Technology Research Centre of Ontario, the Natural Sciences and Engineering Research Council of Canada, and the University of Toronto. Michael Langer's work was partially supported by grants from NSERC and AFOSR, while he was with the Center for Intelligent Machines, McGill University. The authors would like to thank David Waltz and the reviewers for helpful comments.

References

[1] M. Bern, D. Dobkin, D. Eppstein, and R. Grossman. Visibility with a moving point of view. *Algorithmica*, 11:360–378, 1994.

[2] M. Cohen and D. Greenberg. The Hemi-Cube: A radiosity solution for complex environments. In B. A. Barsky, editor, *Computer Graphics (*SIGGRAPH *'85)*, volume 19, pages 31–40, August 1985.

[3] R. Cole and M. Sharir. Visibility problems for polyhedral terrains. *Journal of Symbolic Computing*, 7:11–30, 1989.

[4] Z. Gigus and J. Malik. Computing the aspect graph for the line drawings of polyhedral objects. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 12(2), February 1990.

[5] B. K. P. Horn. *The Psychology of Computer Vision*, chapter Obtaining Shape from Shading Information. McGraw-Hill, New York, 1975.

[6] B. K. P. Horn and R. W. Sjoberg. Calculating the reflectance map. *Applied Optics*, 18(11):1770–1779, 1979.

[7] B.K.P. Horn and M.J. Brooks, editors. *Shape from Shading*. MIT Press, Cambridge, Mass, 1989.

[8] J. J. Kœnderink and A. J. van Doorn. The singularities of the visual mapping. *Biological Cybernetics*, 24:51–59, 1976.

[9] M. S. Langer and S. W. Zucker. Diffuse shading, visibility fields, and the geometry of ambient light. In *Proceedings of the 4th International Conference on Computer Vision*, pages 138–147, Berlin, Germany, 1993.

[10] M. S. Langer and S. W. Zucker. Shape-from-shading on a cloudy day. *Journal of the Optical Society of America A*, 11(2):467–478, 1994.

[11] P.H. Moon and D.E. Spencer. *The Photic Field*. MIT Press, Cambridge, Ma., 1981.

[12] H. Plantinga and C.R. Dyer. Visibility, occlusion, and the aspect graph. *International Journal of Computer Vision*, 5(2):137–160, 1990.

[13] R. Siegel and J.R. Howell. *Thermal Radiation Heat Transfer*. Hemisphere Publ. Corp., 1981.

[14] A. J. Stewart and M. S. Langer. Towards accurate recovery of shape from shading under diffuse lighting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 411–416, June 1996.

[15] G. Ward. The radiance lighting simulation and rendering system. *Computer Graphics (*SIGGRAPH *'94)*, pages 459–471, July 1994.