A LOOPY BELIEF PROPAGATION APPROACH TO THE SHAPE FROM SHADING PROBLEM

Markus Louw, Fred Nicolls
Dept. Electrical Engineering, University of Cape Town, South Africa
markus.louw@gmail.com, fnicolls@dip.ee.uct.ac.za

Dee Bradshaw
Dept. Chemical Engineering, University of Cape Town, South Africa
db@chemeng.uct.ac.za

Keywords: Loopy Belief Propagation, Bayesian network, Shape from shading, photoclinometry.

Abstract: This paper describes a new approach to the shape from shading problem, using loopy belief propagation which is simple and intuitive. The algorithm is called Loopy Belief Propagation Shape-From-Shading (LBP-SFS). It produces reasonable results on real and synthetic data, and surface information from sources other than the image (eg range or stereo data) can be readily incorporated as prior information about the surface elevation at any point, using this framework. In addition, this algorithm proves the use of linear interpolation at the message passing level within a loopy Bayesian network, which to the authors' knowledge has not been previously explored.

1 INTRODUCTION

The interested reader is referred to two surveys, (R. Zhang and Shah, 1999), and (Jean-Denis Durou, 2004). In (R. Zhang and Shah, 1999), SFS approaches are classified into minimization e.g. (Szeliski, 1994), propagation e.g. (S. Osher, 1988), local e.g. (Pentland, 1984), or linear e.g. (P.S. Tsai, 1994) approaches. In (Jean-Denis Durou, 2004), SFS methods are classified into methods based on partial differential equations (characteristic strips (B.K.P.Horn, 1975), power series expansion (Bruss, 1982), and viscosity solutions e.g. (M. G. Crandall, 1983)), minimization methods (P. Daniel, 2000), and "methods approximating the image irradiance equation", which contain the local and linear methods surveyed in (R. Zhang and Shah, 1999).

These surveys describe the development of shape from shading methods, in which researchers have tried to mimic the way the human brain and eyes extract shape information from shading on the object, as well as trying to find analytical solutions based on geometry and reflectance characteristics. This paper describes the casting of the SFS problem into the belief propagation paradigm, which would place it in the minimization and also the propagation class of method. Our method is algorithmically similar to (Jian Sun, 2002), in which Sun et al. use Loopy Belief Propagation (LBP) to solve the dense stereo correspondence problem. In (Jian Sun, 2002), each pixel in the left image is probabilistically assigned disparities for matching to a pixel in the right image, and belief propagation is performed on the nodes which are connected to their immediate (Ising) neighbours. Our method uses a more complicated energy function to approximate the correct elevation map for the surface given the irradiance map, and to enforce surface smoothness conditions. It also incorporates a multi-resolution interpolation based approach which to our knowledge has not used before in Loopy Belief Propagation.

2 Lambertian Lighting Model

This algorithm calculates a surface on the Lambertian assumption that the intensity of a pixel is proportional to the inner product of the direction vector of the incident light and the surface normal at that point. We may follow the notation of (Jean-Denis Durou, 2004), to formulate this. The image irradiance equation is

$$R(\overrightarrow{n}(x)) = I(x) \tag{1}$$

where $I(x)$ is the image irradiance (usually the intensity) measured at location x, and $R(\overrightarrow{n}(x))$ is the reflectance function on the surface which takes the normal at point x as an argument. The surface normal may be calculated as

$$\frac{1}{\sqrt{(1+p(x)^2+q(x)^2)}}(-p(x),-q(x),1) \quad (2)$$

where

$$p = \partial u/\partial x_1 \quad (3)$$

and

$$q = \partial u/\partial x_2 \quad (4)$$

where u is the height of the surface. If there is a unique light source at infinity, and shining in direction $\overrightarrow{w} = (w1, w2, w3)$, the pixel intensity is the inner product

$$R(\overrightarrow{n}(x)) = \overrightarrow{w} \cdot \overrightarrow{n}(x) \quad (5)$$

Hereafter, without loss of generality (but assuming all surface points are visible to both camera and light source) we assume the light source is in the same direction as the camera, which produces an orthogonal projection.

3 Loopy Belief Propagation

The Loopy Belief propagation algorithm using factor nodes may be expressed in the following equations (following loosely the notation of (Murphy, 2002)):

$$\mu_{x\to f}(x) = o(x)\prod_{g\neq f}\mu_{g\to x}(x) \quad (6)$$

$$\mu_{f\to x}(x) = \sum_{u\setminus x} f(u)\prod_{y\neq x}\mu_{y\to f}(y) \quad (7)$$

where x and y are the variable nodes, f and g are factor nodes, $o(x)$ is the prior probability (observation) on the variable node x, and where it is assumed μ is in the domain of f. Our algorithm uses a parallel updating scheme, with the max product algorithm. After the specified number of update iterations, the posterior distribution on each corner vertex node may be given as:

$$p_x(x) = o(x)\prod_{g}\mu_{g\to x}(x), \quad (8)$$

where g is the set of factor node neighbours of x.

4 Formulation of LBP to solve SFS

This algorithm calculates a posterior for the height at each corner vertex on the image. A corner vertex occurs at the corner of a pixel; at the intersection of four pixels, one corner vertex represents the height of the surface at that location. Our energy functional is evolved using Factor nodes which represent the probability of triplets of these corner vertex nodes. Each triplet of vertices creates a unique plane, and the orientation of that plane relative to the direction of the light source allows a probability to be assigned to that configuration for that triplet. If the dimensions of the image are width w and height h and if only simple right angled triangles are used, with the topology shown in Fig. 1, the number of vertices is $(h+1)(w+1)$ and the number of factor nodes is $4wh$. If smoothing using point triplets (described later) is incorporated, the number of Factor nodes is $8wh$. A diagram of the topology for this scheme is shown in Fig.1. The plane generated by each triplet of corner nodes forms an angle against the incident light, giving an illumination for that pixel. This is shown in Fig. 3. Next we define a Markov Random Field (MRF) on this set of vertex nodes X, given the image data Y and explicit range data (which gives a prior probability for the height of the surface at a particular location on the surface):

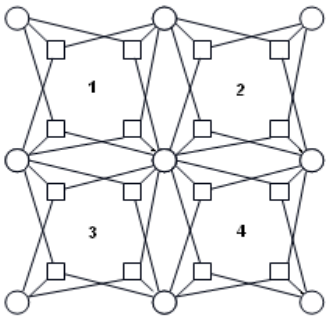

Figure 1: Depiction of the connectivity between vertex nodes (round) and factor nodes (square). These four vertex nodes correspond to the heights of the four corners of a single pixel.

$$P(X|Y,Z) \propto \prod_{i,j,k:k>j>i} \exp -\psi^t_{ijk}(x_i,x_j,x_k,y_{ijk})\cdots \prod_i \exp -\psi_i(x_i,z_i) \quad (9)$$

Each element of the state vector of a vertex node corresponds to the vertex node taking on a particular height. At each iteration, our implementation of Eqn. 7 is the maximum product (aka max. prod. algorithm) of the input messages with the elements of factor node u. A factor node is therefore a 3D array which contains probabilities, each element is derived from the energy

$$\psi^t_{ijk}(x_i,x_j,x_k,y_i) \propto |y_{ijk} - |\overrightarrow{n}_{ijk}(x_i,x_j,x_k)\cdot\overrightarrow{L}||, \quad (10)$$

where N is the number of images (one for each lightsource), $n_{ijk}(x_i,x_j,x_k)$ is the normal of the surface in the triangle between i,j,k, y_{ijk} is the average image intensity in the image region enclosing vertex nodes i,j,k. $\overrightarrow{L}$ is the lightsource direction, x_i is the height of the vertex at right angles to the other two, x_j is the height of the vertex horizontal to that vertex, and x_k is the height of the remaining vertex. We thus have the following equations for the partial derivatives in the height (with respect to change in position in the horizontal and vertical directions on the image) in terms of the three heights of the surface at the points on which the three corner vertex nodes lie:

$$p = \partial u/\partial x_1 = (x_j - x_i)/\partial x_1 \quad (11)$$

$$q = \partial u/\partial x_2 = (x_j - x_i)/\partial x_2. \quad (12)$$

Assuming an affine camera with square pixels and an overall scale of one unit per pixel width we set ∂x_1 and ∂x_2 to 1. We can then use Eqn. 2 to calculate the plane.

Note that it is simple at this stage to extend the energy function to include static scene/moving lightsource information (on the assumption that all points on the surface are always visible to both the camera and to all lightsources). We therefore adjust Eqn. 10 above to be:

$$\psi^t_{ijk}(x_i,x_j,x_k,\overrightarrow{y}_{ijk}) \propto \sum_{s=1}^{N} |y_{ijks} - |\overrightarrow{n}_{ijk}(x_i,x_j,x_k)\cdot\overrightarrow{L_s}||, \quad (13)$$

Where N is the number of images (one for each lightsource), n_{ijk} is the normal of the surface in the triangle between i,j,k, y_{ijks} is the average image intensity in the image region enclosing vertex nodes i,j,k, in image s (note that we have vectorized $\overrightarrow{y}_{ijk}$ to show that it contains average intensity values over all intensity images, indexed with the subscript s. So s iterates over each of the images, and $\overrightarrow{L_s}$ is the lightsource direction in image s.

4.1 Boundary conditions and range data

In Shape from Shading algorithms, it is usually necessary to establish some boundary conditions, since all surface heights calculated (if only shape from shading be used) are relative to each other, but not against any fixed frame of reference. In addition, the specification of boundary conditions may solve some of the ambiguities, since there are usually a number of surfaces which may generate a particular intensity map under particular lighting conditions. LBP allows such boundary conditions and range data to take on the form of either hard or soft constraints, and this is handled naturally within the LBP paradigm. Each corner vertex node x_i may be given a prior probability on the heights of its state vector, such that

$$x_i(X = h) \propto \exp(-|h - u(x_i)|), \quad (14)$$

where $u(x_i)$ is the specified range or depth of the point. Whether the point is given a value because it lies on a known boundary, or because we have range data about the point, the point is treated the same way.

4.2 Multi-Resolution

The LBP method allows us to approach the problem in a multi-resolution manner: after N LBP iterations, we may iterate through each of the M corner vertex nodes and adjust the heights which each element in the vertex node's state vector corresponds to, so that we may home in on a closer approximation of the correct value. The multi-resolution algorithm is described as follows:

1. for i = 1 to N
2. for j = 1 to M
3. calculate MAP estimate for all corner vertex node
4. end
5. increase height resolution of each corner vertex node
6. end

In step 5 in the above algorithm, for each corner vertex node, we calculate the posterior on that node using Eqn. 8. Then, we find the entry with the highest probability. The new set of heights for that node are then calculated using a finer height resolution about the MAP value. The algorithm for calculating the new set of heights is

1. Calculate new height resolution: $h^{new} = kh^{old}$
2. $n \leftarrow 1$; $a \leftarrow 1$; $b \leftarrow 1$
3. while ($n < numlabels$)
4. if ($LB < H_x + a \cdot h^{new} < UB$)
5. vertex height(n) $\leftarrow H_x + a \cdot h^{new}$
6. $n \leftarrow n+1$; $a \leftarrow a+1$
7. end
8. if ($n < numlabels$)
9. if ($LB < H_x - b \cdot h^{new} < UB$)
10. vertex height(n) $\leftarrow H_x - b \cdot h^{new}$
11. $n \leftarrow n+1$; $b \leftarrow b+1$
12. end
13. end
14. end

where in the above algorithm, H_x is the MAP estimate for the height at x, h^{new} is the height resolution for the current LBP resolution, k is a compression ratio applied to the previous height resolution h^{old} at each resolution cycle, LB and UB are the lower and upper bounds respectively. We then update the messages from the corner vertex nodes to the factor nodes to reflect the new heights of the state vector of the corner vertex nodes, using linear interpolation between the points on the original values.

$$\mu^{new}_{x\to f}(x) = L(\mu^{old}_{x\to f}(x), \mathbf{h}^{new}, \mathbf{h}^{old}), \quad (15)$$

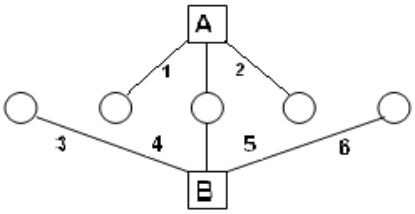

Figure 2: Factor node connectivity for first and second order smoothing on collinear triplets of corner vertex nodes. The corner nodes are circles and factor node squares. Factor node "A" implements smoothing over a small scale, while factor node "B" implements smoothing over a larger scale. The numbers correspond to pixels (there is one pixel interior to four corner vertices).

with L the linear interpolation function, and $\mathbf{h}^{new}, \mathbf{h}^{old}$ are the height labels at the new and old resolutions, respectively. Linear interpolation at point b between two points a and c given function $f(\cdot)$ is given by $f(b) = f(a) + (b-a)(f(c)-f(a))/(c-a)$ or $f(b) = f(c) - (c-b)(f(c)-f(a))/(c-a)$.

4.3 Smoothing

If the energy terms used is that for the triangle topology used are simply those shown in Fig. 1 and written in Eqn. 9, it is likely that the algorithm may converge to a solution (digital elevation map) with undesirable high frequency components.

4.3.1 Smoothing with triangles of varying size

Undesirable high frequency characteristics may be reduced using larger triangles (i.e. with the same kind of factor node entries, but with larger areas, and with the intensity over the surface of each triangle averaged over its surface), as is shown in Fig. 4.

4.3.2 Smoothing with collinear point triplets

An alternative is to form an energy term over all collinear (adjacent) point triplets on the height map. The smoothing term should correspond to the true nature of the surface as closely as possible. If the surface is known to have only low frequency spatial change in height, then the smoothing energy term should penalize rapid height variation. The topology of the Bayesian network w.r.t. vertex and factor nodes for smoothing is shown in Fig. 2. Two factor nodes are depicted, to show how different smoothing energy terms may be applied at different scales.

The energy function associated with the smoothing of the corner vertex node triplets is (cf. Eqn. 10)

$$\psi^s_{ijk}(x_i,x_j,x_k) = \exp\left(\left(\frac{h(x_j)-h(x_i)}{d(i,j)} - \frac{h(x_k)-h(x_j)}{d(j,k)}\right)^2/\sigma\right), \quad (16)$$

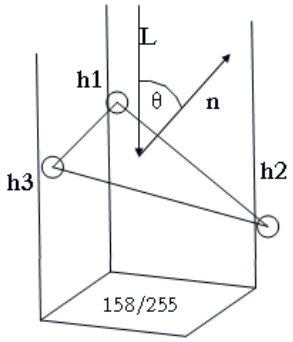

Figure 3: Depiction of the plane generated by corner vertex nodes each at a particular height. The inner product of the plane's normal and the light source's direction gives the pixel intensity at the pixel corresponding to those three corner vertex nodes, which in this diagram is 158/255 .

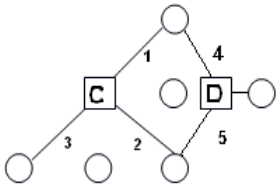

Figure 4: The connectivity of an isosceles triangle (factor node "D"), and a large right angled triangle (factor node "C"). The factor nodes are square, the corner nodes are round, and the energy terms for each triangle are represented in the factor nodes.

where $h(\cdot)$ is the height of a corner vertex node for a particular value, and $d(i,j)$ is the distance between the two corner vertex nodes i and j. It is assumed in this equation that point j lies between i and k, and the three points are collinear. The smoothness may be adjusted through σ. With this smoothing energy term, we can rewrite Eqn.9 as

$$P(X|Y,Z) \propto \prod_{i,j,k:k>j>i} \psi^t_{ijk}(x_i,x_j,x_k,y_i)\cdots \psi^s_{ijk}(x_i,x_j,x_k)\prod_i \psi_i(x_i,z_i) \quad (17)$$

5 Results

We tested the algorithm on real and synthetic data: the synthetic data was created by generating different smooth 3D shapes, and calculating the image of those shapes under the Lambertian model. The light source was assumed to be in the same direction as the camera, and the image projection of the surface orthogonal. For real data we used images of individual froth bubbles from a mineral ore flotation cell.

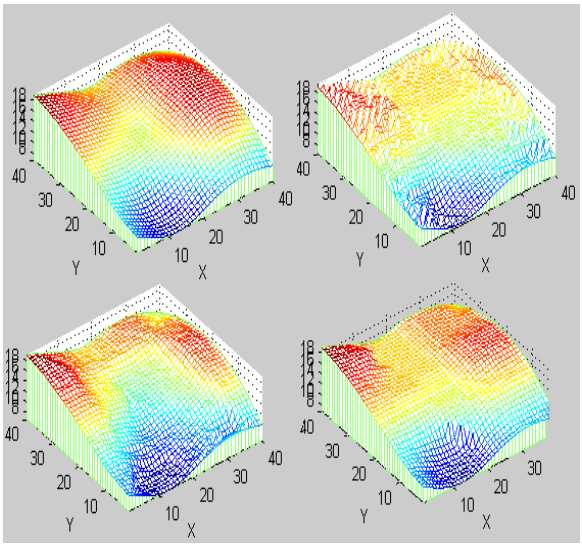

Figure 5: A good run using LBP-SFS. In this example 26 labels per vertex node were used, and 7 Resolutions were done with a compression ratio of 0.8, with 200 iterations per resolution. The top left surface is the true one, the top right surface is LBP-SFS run using a single reflectance map, the bottom left uses 2 reflectance maps, and the bottom right surface uses 3 reflectance maps. More information allows for improved surface calculation.

5.1 Synthetic Data

For this synthetic data, we generate some smooth surfaces and supply a lighting direction and camera position to calculate the intensity map of the image under Lambertian assumptions. Examples of the synthetic surfaces are shown in Fig. 5.

5.2 Real Data

For the real data we show the 3D models along with the images of the bubbles on which LBP-SFS was run. The algorithm was run at various height resolutions, and the run times at each of the resolutions is shown. In the real images, noise was removed by applying convolution with a Gaussian kernel

6 Discussion

The algorithm has advantages and disadvantages. The main problem is that is is prone to fall into local minima which are incorrect, resulting in deformation in the reconstruction (e.g. Fig. 7). Even after the boundary conditions and some range data have been given, there are many different surfaces which could produce the observed image intensity data. Ways of overcoming this within the LBP-SFS paradigm include adding a smoothing energy term on

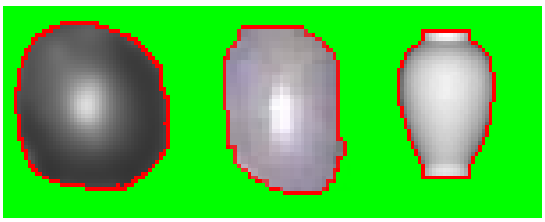

Figure 6: Images of two bubbles and a vase with overhead lighting. The circumference of each object is chosen as a zero level boundary condition for the bubble. (This is smoothed with a Gaussian kernel before LBP-SFS is applied).

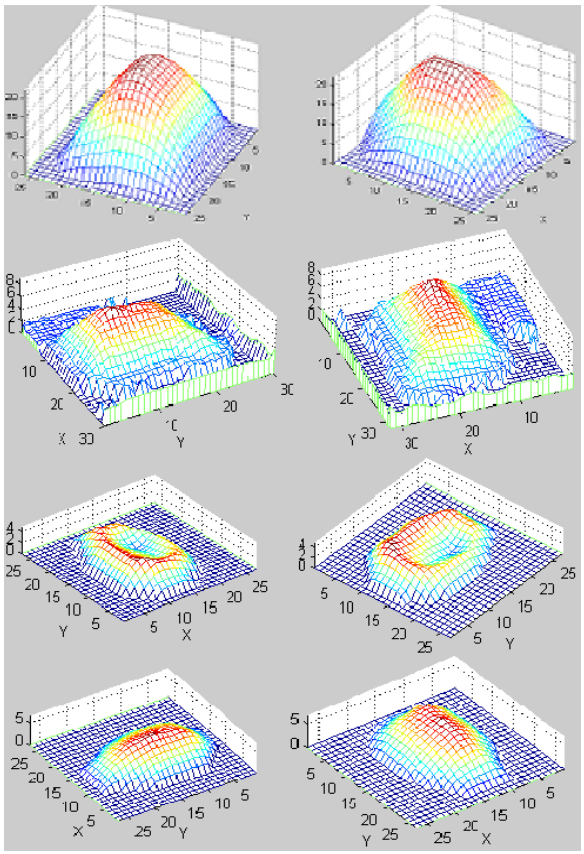

Figure 7: A LBP-SFS reconstruction of the objects shown in in Fig. 6, assuming Lambertian reflectance. The first and second bubble (first and second row surfaces respectively) had no surface points given. The surfaces in the third row show the reconstructed vase with no surface points given, and in the last row it is shown with a single surface point given. Energy terms used were those corresponding to small triangles (Fig. 1), isosceles triangles (Fig. 4) and first order smoothing (Fig. 2) with $\sigma = 1$.

collinear point triplets and the use of surface triangles of varying scales to eliminate high frequency error in the resultant digital elevation map. Other methods e.g. (P.L. Lions, 1993) usually remove some ambiguity by defining a single highest point or characteristic curve (M. G. Crandall, 1983),(M. G. Crandall, 1984),(E. Rony, 1992), (S. Osher, 1988). Our algorithm may be made more reliable by increasing the number of labels used to represent corner vertex heights, although each extra label greatly increases the computation time. We have also noted that elevation map results using this algorithm are more reliable for smaller images (less than 50×50). As the image size increases, the algorithm is more prone to fall into bad local minima. We are experimenting with a wavelet-style spatial multi-resolution approach to overcome this. The algorithm results will improve with an increase in the number of soft range data points supplied, and with increased certainty on those points. This method should really be seen as a data fusion method for incorporating range data with intensity information.

7 Conclusion

The algorithm has the advantage of being adjustable in terms of the height resolution required per vertex. Unlike other SFS methods, LBP-SFS can incorporate both hard and soft constraints on the boundary conditions of the surface and range data at points on the surface. Different reflectance models per surface can be easily accounted for in the energy term. This algorithm requires long computation time and large storage for images with large depth variation (such images would require larger vertex node state vectors given an initial height resolution).

8 Current and Future Work

Preliminary results show that minimizing the same MRF formulations using simulated annealing with Gibbs sampling is faster and more reliable. We are also investigating integrating SFS information into a dense stereo formulation.

9 Acknowledgements

The authors are grateful for the financial support given by the National Research Foundation of South Africa, and Anglo American via the Minerals Processing Research Unit at the University of Cape Town.

REFERENCES

B.K.P.Horn (1975). *Obtaining Shape from Shading Information*. McGraw-Hill.

Bruss, A. (1982). The eikonal equation: Some results applicable to computer vision. *Journal of Mathematical Physics*, pages 890–896.

E. Rony, A. T. (1992). A viscosity solutions approach to shape-from-shading. *SIAM. J. Numer. Anal.*, pages 867–884.

Jean-Denis Durou, Maurizio Falcone, M. S. (2004). Numerical methods for shape from shading: A survey with benchmarks. *CVIU*.

Jian Sun, Heung-Yeung Shum, N.-N. Z. (2002). Stereo matching using belief propagation. *ECCV*.

M. G. Crandall, P. L. (1983). Viscosity solution of hamilton-jacobi equationssfs. *Trans. Amer. Math. Soc.*, pages 1–42.

M. G. Crandall, P. L. (1984). Two approximations of solutions of hamilton-jacobi equations. *Math. Comput.*, pages 907–922.

Murphy, K. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley.

P. Daniel, J.-D. D. (2000). *From Deterministic to Stochastic Methods for Shape from Shading*. In Proc. 4th Asian Conf. on Comp. Vis.

Pentland, A. (1984). Local shading analysis. *IEEE. transactions on Pattern Analysis and Machine Intelligence*, pages 170–187.

P.L. Lions, E. Rouy, A. T. (1993). Shape-from-shading, viscosity solutions and edges. *Numerische Mathematik 64*, pages 323–353.

P.S. Tsai, M. S. (1994). Shape from shading using linear approximation. *Image and Vision Computing Journal*, pages 187–198.

R. Petrovic, I. Cohen, B. F. R. K. and Huang, T. (2001). Enforcing integrability for surface reconstruction algorithms using belief propagation in graphical models. *CVPR*.

R. Zhang, P. Tsai, J. C. and Shah, M. (1999). Shape from shading: A survey. *PAMI*.

S. Osher, J. S. (1988). Fronts propagating with curvature-dependent speed: algorithms based on hamiltonian-jacobi formulations. *J. Comput. Phys.*, pages 12–49.

Szeliski, R. (1994). Fast shape from shading. *Computer Vision, Graphics, Image Processing: Image Understanding*, pages 129–153.