Proceedings of the 1998 IEEE International Conference on Computer Vision, Bombay, India

Depth Discontinuities by Pixel-to-Pixel Stereo *

Stan Birchfield
Department of Electrical Engineering
Stanford University
Stanford, California 94305
birchfield@cs.stanford.edu

Carlo Tomasi
Department of Computer Science
Stanford University
Stanford, California 94305
tomasi@cs.stanford.edu

Abstract

An algorithm to detect depth discontinuities from a stereo pair of images is presented. The algorithm matches individual pixels in corresponding scanline pairs while allowing occluded pixels to remain unmatched, then propagates the information between scanlines by means of a fast postprocessor. The algorithm handles large untextured regions, uses a measure of pixel dissimilarity that is insensitive to image sampling, and prunes bad search nodes to increase the speed of dynamic programming. The computation is relatively fast, taking about 1.5 microseconds per pixel per disparity on a workstation. Approximate disparity maps and precise depth discontinuities (along both horizontal and vertical boundaries) are shown for five stereo images containing textured, untextured, fronto-parallel, and slanted objects.

1 Introduction

Cartoon artists have known the perceptual importance of depth discontinuities for a long time. To create the illusion of depth, they paint the character and background on different layers of acetate, being careful to ensure a crisp delineation of the character. Similarly, in human stereo vision, depth discontinuities are vividly perceived and help to carve out distinct objects as well as to elucidate the distance relations between them.

In this paper we present a method for detecting depth discontinuities from a stereo pair of images. Our approach inverts the traditional role of a stereo algorithm because, instead of using the knowledge of depth discontinuities to compute disparity more accurately, we compute a rough disparity map in order to get crisp discontinuities. Like several previous algorithms [2, 5, 7, 8], our algorithm uses a form of dynamic programming to match epipolar scanlines independently, detecting occlusions and depth discontinuities simultaneously with a disparity map. Then a postprocessing step propagates information between the scanlines to refine the disparity map and the depth discontinuities. Throughout the process, we use neither windows nor preprocessing of the intensities, thus matching the individual pixels in one image with the pixels in the other image.

As a stereo algorithm, our approach contains three novelties. First, the image sampling problem is overcome by using a measure of pixel dissimilarity that is insensitive to sampling. Secondly, the algorithm handles large untextured regions which present a challenge to many existing stereo algorithms. Finally, unlikely search nodes are pruned to reduce dramatically the running time of dynamic programming. The combination of avoiding subpixel resolution, pruning bad nodes, and fast postprocessing results in an efficient algorithm that takes 1.5 microseconds per pixel per disparity on a workstation, making it a candidate for real-time implementation.

2 Stereo Formulation

In this section, we formulate the stereo problem and describe our cost function. Pixels in one image are explicitly matched with pixels in the other image, while occluded pixels remain unmatched. Correspondence is encoded in a *match sequence*, where each *match* is an ordered pair (x, y) of pixels signifying that the intensities $I_L(x)$ and $I_R(y)$ are images of the same scene point. (Throughout this paper, x denotes a pixel in the left scanline, while y denotes a pixel in the right scanline.) Unmatched pixels are *occluded*, and a subsequence of adjacent occluded pixels that is bordered by two non-occluded pixels (or by a non-occluded pixel and the image boundary) is called an *occlusion*.[1] An example of a match sequence on an extremely short scanline is shown in Figure 1.

The *disparity* $\delta(x)$ of a pixel x in the left scanline that matches some pixel y in the right scanline is defined in the usual way as $x - y$, while the disparities of all the pixels in an occlusion are assigned the disparity of the farther of the two neighboring objects. The *depth-discontinuity pixels* are labelled as those pixels that border a change of at least

* Work supported by grants NSF IRI-9506064, ARO-MURI DAAH04-96-1-0007 and ARO STTR F49620-95-C-0078, by an NSF Graduate Student Fellowship, and by a gift from the Charles Lee Powell Foundation.

[1] Our *occlusion*s correspond roughly to Belhumeur's *half-occluded regions* [2].

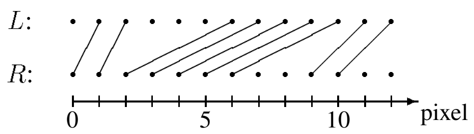

Figure 1: The match sequence $M = \langle(1,0)$, (2,1), (6,2), (7,3), (8,4), (9,5), (10,6), (11,9), $(12,10)\rangle$. The five middle matches correspond to a near object.

two levels of disparity and that lie on the far object. At the expense of losing some true depth discontinuities, this threshold of two allows us to handle slanted objects without explicitly detecting the slant.

2.1 Cost function

With each match sequence M we associate a cost $\gamma(M)$ that measures how unlikely it is that M describes the true correspondence. Instead of deriving a maximum a posteriori (MAP) cost function from a Bayesian formulation (as is done in [2, 7, 11]), we propose a simple cost function justified solely by empirical evidence.

The cost of a match sequence is defined by a constant penalty for each occlusion, a constant reward for each match, and a sum of the dissimilarities between the matched pixels:

$$\gamma(M) = N_{occ}\kappa_{occ} - N_m\kappa_r + \sum_{i=1}^{N_m} d(x_i, y_i), \qquad (1)$$

where κ_{occ} is the constant occlusion penalty, κ_r is the constant match reward, $d(x_i, y_i)$ is the dissimilarity between pixels x_i and y_i, and N_{occ} and N_m are the number of occlusions (not the number of occluded pixels) and matches, respectively, in M.

This cost function prefers piecewise-constant disparity maps. Thus, if possible, each object is assigned a single disparity, even if that object's depth varies in actuality (as in the case of a cylindrical surface). Although this behavior sacrifices accurate scene reconstruction, it facilitates the precise localization of depth discontinuities because it accentuates the change in disparity at the object's boundaries (at least in case of objects, like cylinders, whose depth tapers at the ends). In addition, the simplicity of (1) makes our cost function easy to understand, implement, and evaluate.

2.1.1 Occlusion penalty and match reward

Technically, κ_{occ} is interpreted as the amount of evidence (in terms of mismatched pixel intensities) that is necessary to declare a change in disparity, while κ_r is interpreted as the maximum amount of pixel dissimilarity that is generally expected between two matching pixels. Together, the two terms act like an occlusion penalty that is dependent on the length of the occlusion [2, 7]. Nevertheless, we keep the terms separate because a constant occlusion penalty is central to our method of pruning the search space, as described in Section 3.2. In our implementation, $\kappa_{occ} = 25$ and $\kappa_r = 5$ (both measured in gray levels).

2.1.2 Pixel dissimilarity

The term $d(x_i, y_i)$ measures how unlikely it is that $I_L(x_i)$ and $I_R(y_i)$ are images of the same scene point. This dissimilarity cannot be measured by simply taking the difference between $I_L(x_i)$ and $I_R(y_i)$, as is often done, because image sampling can cause this difference to be large in the vicinity of intensity edges. Typically, the problem is alleviated either by working at subpixel resolution [2, 11] or by adding robustness through window-based matching [4, 6, 7, 9]. But subpixel resolution is computationally expensive for algorithms that explicitly search over all possible disparities, and windows degrade the precision of the depth discontinuities since depth discontinuities violate the fundamental assumption behind windows. Therefore, we propose instead to use the linearly interpolated intensity functions surrounding two pixels to measure their dissimilarity, in a method that is provably insensitive to sampling.

To understand our dissimilarity measure in more detail, consult Figure 2, which shows the intensity functions I_L and I_R incident upon two corresponding scanlines of the left and right cameras, respectively. The functions are sampled at discrete points by the image sensor; three such adjacent points (or pixels) are shown here in each scanline. In this discussion, x_i and y_i are chosen as the pixels whose dissimilarity is to be measured. We define $\hat{I}_R$ as the linearly interpolated function between the sample points of the right scanline. Then we try to measure how well the intensity at x_i fits into the linearly interpolated region surrounding y_i. That is, we define the following quantity:

$$\bar{d}(x_i, y_i, I_L, I_R) = \min_{y_i - \frac{1}{2} \le y \le y_i + \frac{1}{2}} |I_L(x_i) - \hat{I}_R(y)|.$$

Then, the dissimilarity between the pixels is computed as the minimum of this quantity and its symmetric counterpart:

$$d(x_i, y_i) = \min\{\bar{d}(x_i, y_i, I_L, I_R),\ \bar{d}(y_i, x_i, I_R, I_L)\}.$$

Thus, the definition of d is symmetrical.

Since the extreme points of a piecewise linear function must be its breakpoints, the computation of d is rather straightforward. Again, see Figure 2. First we compute $I_R^- \equiv \hat{I}_R(y_i - \frac{1}{2}) = \frac{1}{2}(I_R(y_i) + I_R(y_i - 1))$, the linearly interpolated intensity halfway between y_i and its neighboring pixel to the left, and the analogous quantity $I_R^+ \equiv$

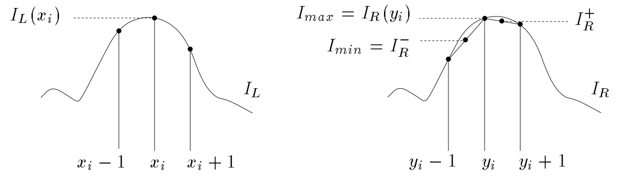

Figure 2: Definition & computation of $\bar{d}(x_i, y_i, I_L, I_R)$.

$\hat{I}_R(y_i + \frac{1}{2}) = \frac{1}{2}(I_R(y_i) + I_R(y_i + 1))$. Then we let $I_{min} = \min(I_R^-,\ I_R^+,\ I_R(y_i))$ and $I_{max} = \max(I_R^-,\ I_R^+,\ I_R(y_i))$. With these quantities defined,

$$\bar{d}(x_i, y_i, I_L, I_R) = \max\{0, I_L(x_i) - I_{max}, I_{min} - I_L(x_i)\}.$$

This computation takes only a small, constant amount of time more than the absolute difference in intensities.

The quantity d is insensitive to sampling in the sense that, without noise or other distortions, $d(x_i, y_i) = 0$ whenever y_i is the closest sampling point to the y value corresponding to x_i. The only restriction is that the continuous intensity function incident upon the sensor be either concave or convex in the vicinity of x_i and y_i (Interested readers can find the theorems and proofs in [3]). In practice, inflection points cause no problem since the regions surrounding them are approximately linear — and linear functions are both concave and convex. Therefore, our cost function works well as long as the intensity function varies slowly compared to the pixel spacing on the sensor, i.e., as long as aliasing does not occur. We slightly defocus the lens to ensure this condition.

Figure 3 contrasts our dissimilarity measure with the absolute difference in intensity. Wherever the intensity function is nearly constant, or wherever the disparity between the two scanlines is close to an integral number of pixels, the two approaches yield similar results, since sampling effects are negligible. In the remaining areas, however, the absolute difference can be large, while our measure remains well-behaved.

2.2 Hard constraints

In addition to measuring the likelihood of a match sequence by its cost, we require all match sequences to satisfy certain constraints. The first set of constraints enables the algorithm to handle untextured regions, while the second set facilitates a systematic, efficient search.

2.2.1 Intensity variation accompanies depth discontinuities

Because of the ambiguity in untextured regions, many stereo algorithms require texture throughout the images. In fact, it is not uncommon for a scene to be artificially altered by placing a textured background behind the objects of interest in order to make the scene more amenable to the particular stereo algorithm being tested. As we will demonstrate, however, untextured, nearly fronto-parallel surfaces can be handled quite nicely as long as one assumption remains true, namely that intensity variation accompanies depth discontinuities.[2] (Similar assumptions have been used in [4, 6].) Because our threshold of declaring intensity variation is small, we are not trying to place the depth discontinuities along strong intensity "edges" but are merely preventing the cost function from making a poor decision in a region with no information.

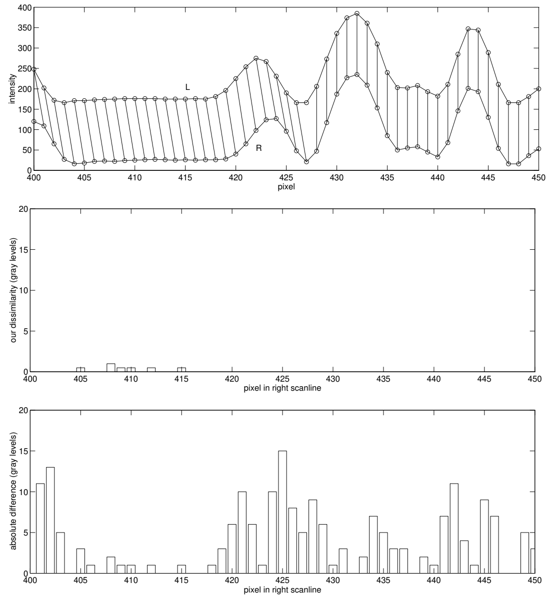

Figure 3: TOP: A portion of a match sequence. For viewing clarity, the left scanline is shifted up, while the right scanline is shifted to the right. MIDDLE: The dissimilarities between the matched pixels, as computed by our measure. Most of the values are zero. BOTTOM: The dissimilarities computed by taking the absolute value of the difference in intensity.

Previous algorithms have not exploited the full potential of this assumption. Not only does the assumption constrain a depth discontinuity to lie *near* intensity variation, but it also specifies upon which side of the variation the discontinuity must lie. To see this, note that intensity variation occurring at a depth discontinuity (as the result of an intensity difference between the near object and the far object) has the same disparity as the near object (see Figure 4). This fact is not hard to see once one realizes that the physical origin of the intensity variation is the boundary of the near object, regardless of the geometry of the far object. There-

[2] An *intensity variation* is declared roughly as follows: any set of three adjacent pixels whose difference between maximum and minimum gray levels is at least five. Think of it as a weak intensity edge.

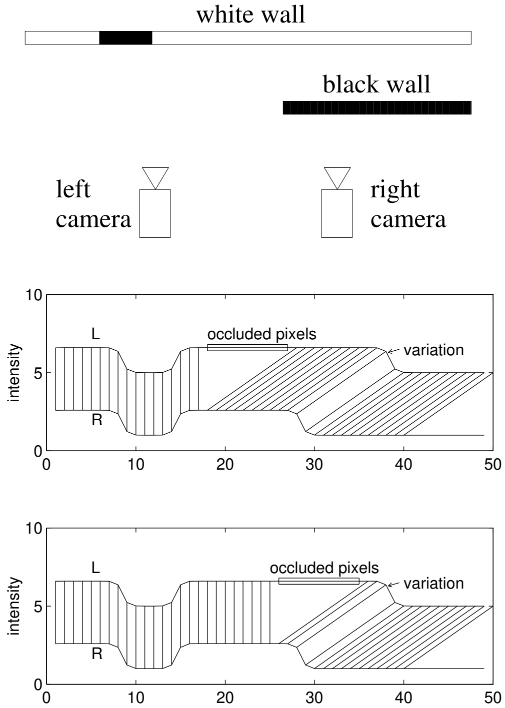

Figure 4: Given the assumption that there is a change in intensity along the boundary between the near and far objects, intensity variation must lie to the right of an occlusion in the left scanline. TOP: A physical setup. MIDDLE: A match sequence that appears feasible but actually violates the assumption. BOTTOM: The match sequence that is consistent with the assumption.

fore, as the camera moves laterally, the intensity variation moves with the projection of the near object. Using Figure 4 as an example, we notice that pixels in the left scanline are occluded when the far object's projection is to the left of the near object's. Since the occluded pixels come from the far object, and since the intensity variation is part of the near object, the occlusion must lie immediately to the left of the intensity variation. Likewise, occluded pixels in the right scanline must lie immediately to the right of intensity variation. Therefore, we require each occlusion to be accompanied by intensity variation on the appropriate side.

2.2.2 Constraints related to search

Like most stereo algorithms, we impose a limit on the amount of disparity allowed: $0 \leq \delta \leq \Delta$. Also, to enable the use of dynamic programming we impose the monotonicity constraint and forbid simultaneous left and right occlusions, this latter constraint being equivalent to the requirement that, if (x, y) is a match, either $x + 1$ or $y + 1$ must be matched.

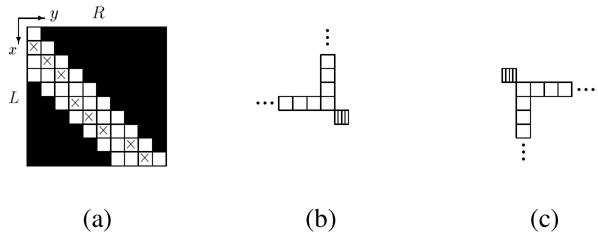

Figure 5: (a) The search grid and a match sequence ("$\times$" cells). (b) The matches (white cells) that can immediately precede a match (striped cell). (c) The matches that can immediately follow a match.

3 Searching Along Epipolar Scanlines

Thanks to the structure of the cost function, the technique of dynamic programming (also used in [1, 2, 5, 7, 8, 12]), can be used to find the optimal match sequence by conducting an exhaustive search.

Figure 5a illustrates the search grid for two scanlines having 10 pixels each, using a maximum disparity of three pixels (i.e., $\Delta = 3$). Because of the disparity limit, many of the cells in the grid are disallowed; these are shown as black cells. The algorithm searches for the best possible path[3] stretching from the left-hand side to the right-hand side. As an example, the match sequence $\langle(1,0),(2,1),(3,2),(5,3),(6,4),(7,5),(8,7),(9,8)\rangle$ is shown by the cells marked with $\times$. Notice that any column or row that does not contain an $\times$ corresponds to an occluded pixel.

3.1 Two dual optimal algorithms

The standard dynamic programming algorithm would find the best path by iterating through all the cells in the search grid, computing the best path to each cell. However, an equivalent algorithm computes the best paths *through* each cell. That is, each time a cell is encountered, the paths through that cell to all its possible following cells are computed. For example, let c be the cell, and let c_f be one of its following cells. Then, if the path to c_f through c is better than any previously computed path to c_f, the path to c_f is updated. Basically, instead of looking backward, as in Figure 5b, each cell looks forward, as in Figure 5c. This concept is tricky to explain but not hard to understand.

Why is this algorithm important, since its computation is identical to that of the standard algorithm? With the standard algorithm, the worth of a cell is not known until after the computation has already been performed for that cell. In contrast, with this alternate algorithm, the best

[3] Informally, we will use the terms *path* and *match sequence* interchangeably, as well as the terms *cell* and *match*.

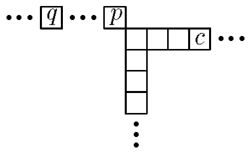

Figure 6: Optimality is retained when p is not rightward expanded, assuming that $\gamma_0(q) < \gamma_0(p)$.

path to each cell is computed *before* the cell is expanded. Therefore, the search space can be pruned by refusing to expand cells unlikely to be along the best path.

3.2 A faster algorithm

Consider a match p with a possible following match c such that there are right-occluded pixels between them, as shown in Figure 6. Now suppose that there is some match q to the left of and on the same row as p whose best path has a lower cost. Then q is also a possible preceding match of c (as is evident from Figure 5c), and the best path to c through q is better than the best path to c through p, since the occlusion penalty is constant. Therefore, there is no need to expand p to c, or indeed to any of the matches on c's row since q is also a possible preceding match of each of them. By a similar argument, we conclude that it is fruitless to expand p to any of the matches on its adjacent column if there is a lower-cost match above it.

In light of these observations the algorithm could, without sacrificing optimality, refuse to rightward expand any match with a lower-cost match to its left or downward expand any match with a lower-cost match above it. However, the running time would not be reduced because of the difficulty in determining whether there is a lower-cost match above or to the left of another match. Instead, the algorithm refuses to rightward expand any match with a lower-cost match in its row or downward expand any match with a lower-cost match in its column. This pruning brings the running time down from $O(n\Delta^2)$, where n is the number of pixels in the scanline, to approximately $O(n\Delta \log \Delta)$, as is evident from Figure 7.

4 Propagating Information Between Scanlines

While processing scanlines independently is computationally attractive and straightforward to formulate, it does not take advantage of the dependence of the disparities from one scanline to the next. A common way to incorporate this information is to extend the one-dimensional cost function to a two-dimensional cost function, which is then minimized. However, minimizing such a function in a computationally efficient manner is not a straightforward

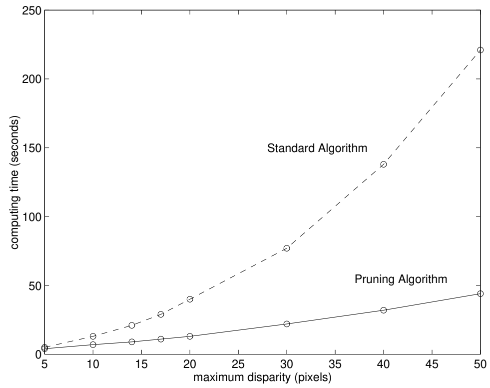

Figure 7: Computing time vs. Δ of our algorithm (solid) and the standard algorithm (dashed).

task. In the extension from 1D to 2D, it is not uncommon for the computing time to increase by 800% or more [2, 12]. As a result, some approaches avoid the extension altogether [7, 8].

We have devised a method for postprocessing the disparity map by propagating reliable disparity values into regions of unreliable disparity values. This postprocessing is rather global in nature and is quite effective at propagating the background disparities into regions with little intensity variation. Moreover, it is fast, increasing our processing time by only 30%.

Each pixel is assigned a level of *reliability*, which is determined by the number of contiguous pixels in the column agreeing on their disparity. The idea is that if the disparities of pixels on adjacent rows were computed independently and they agree, then they are likely to be correct (except in one case to be described shortly). Pixels are quantized into one of three nondisjoint categories (that is, each category subsumes the previous ones): *slightly reliable*, *moderately reliable*, or *highly reliable*. We can think of moderately reliable pixels as being aggressive, changing the values of their neighbors, while slightly reliable pixels are defensive, resisting change.

A moderately reliable pixel propagates along its column, changing the disparities of the pixels it encounters, until it reaches either intensity variation or a slightly reliable region with a lower disparity. Regions with a higher disparity are overrun no matter what their reliability, because reliability is not a good indication that the disparities are correct when the background has little intensity variation. For example, after the initial processing of Figure 8a, all 55 rows incorrectly agree that the lamp's concavity should be assigned the disparity of the lamp.

The only distinction between moderately and highly reliable pixels is that the former are not allowed to overrun their neighbors if the change in disparity is just one pixel.

This rule preserves some slanted surfaces, such as the table and boxes in Figures 8a and 8c.

After the pixels are propagated along their columns, the same process is repeated along the rows. Reliability is determined by the number of contiguous pixels in a row agreeing on their disparity, and disparities are then propagated horizontally. This step has less theoretical justification than the previous one, but it helps to fill in some of the remaining gaps. Before either the vertical or horizontal propagation step has begun, the disparity map is cleaned by removing isolated disparity values that are surrounded by values that agree, and after both propagation steps have finished, the disparity map is cleaned by mode filtering.

5 Experimental Results

We present the results of the algorithm on five stereo pairs, shown in Figure 8. The images were taken with a single Pulnix camera whose lens was slightly defocused to remove aliasing and which was translated along a baseline of 10 mm. The results demonstrate the algorithm's ability to compute an approximate disparity map and accurate depth discontinuities in a wide variety of situations, such as textured and untextured objects, textured and untextured backgrounds, curved and planar surfaces, specular and matte surfaces, and fronto-parallel and slanted surfaces.

Particularly striking is the result in Figure 8a, in which the depth discontinuities are nearly perfect. Notice that the discontinuities are correctly placed along the edges of the table support and the lamp cord, even though the only texture between the two is a little door hinge. Also, the table is recovered as a series of constant-disparity strips whose disparity decreases as the table recedes. Figure 8b shows similar performance, although somewhat more noisy, with a textured background.

The results of Figure 8c are also worth noting. Even though the algorithm generally assumes fronto-parallel surfaces and has no explicit representation of a slanted surface, the depth discontinuities are recovered in the presence of both horizontal and vertical slant.

From these images, it is easy to see both the power and drawback of ignoring one-level disparity transitions in the labelling of depth discontinuities. Although many false transitions are ignored, such as those on the slanted tables and the right box of Figure 8c, some true transitions are improperly forgotten, such as the back edge of the table in Figure 8a. It is important to note that even in principle this problem can never be eliminated completely, because it is impossible to determine the discontinuities of a continuous function from a sampled version.

Probably the main drawback to the algorithm is its brittleness. Because of its emphasis on speed and on preserving sharp changes in disparity, the algorithm is heavily dependent upon local information. For example, if a boundary has no accompanying intensity variation for several scanlines in a row, then that boundary will not be found (see for example the triangular wedge and cap of the left Clorox bottle in Figure 8e). Similarly, moving the lamp of Figure 8b slightly to one side can cause the middle of the lamp post to be assigned the disparity of the background because of the lack of intensity variation at the boundaries. Brittleness also becomes evident with subsampled images or an increased baseline, both of which cause the intensities in the two images to look different.

On these 630×480 images, with the maximum disparity Δ set to 14, a Silicon Graphics Indy workstation took 8 seconds to match the scanlines independently and 2.5 additional seconds for postprocessing. An Indigo 2 Extreme needed 5.5 and 1.5 sec., respectively.

6 Comparison with Previous Work

It is instructive to imagine how other stereo algorithms would handle the image in Figure 8a. Intensity-based algorithms such as those by Belhumeur and Mumford [2], Cox et al. [5], Geiger et al. [7], and Intille and Bobick [8] have no mechanism for preferring to place depth discontinuities near intensity variation and would therefore not place the discontinuities along the contour of the lamp. Moreover, since the latter two methods do not incorporate information between scanlines, they would not fill in the concavity of the lamp or the region between the table support and the lamp cord (It could be argued that the former two methods would fare no better in this respect). For a similar reason, the algorithms of Luo and Burkhardt [11] and Jones and Malik [9] would not be able to find the lamp's boundary.

Although the algorithms of Fua [6] and Cochran and Medioni [4] try to align the depth discontinuities with the intensity edges, it is not clear how well they would perform on this image because the initial disparity map would be so far from the true solution (due to the untextured regions and the algorithms' dependence upon local information for the initial matching).

The methods of Baker and Binford [1] and Ohta and Kanade [12] would probably match the intensity edges correctly, yielding a sparse disparity map. However, in interpolating the disparity of the untextured regions neither method would preserve the sharp depth discontinuities. Moreover, edge detectors have difficulty in dealing with weak edges, such as that of the recorder in Figure 8d (column 190).

Some algorithms directly detect depth discontinuities, without computing dense correspondence. Because the approaches of Little and Gillett [10] and Toh and Forrest [13] use only local information, they would find few if any depth discontinuities in Figure 8a, which contains little texture. Wixson's algorithm [14] is similar to that of Toh and Forrest in that it matches nearly vertical edges in both images by correlating the two regions on either side of the edge. Since an edge must have texture on both sides, the contour along the right side of the lamp would not be found, nor would the nearly horizontal table edge.

7 Conclusion

Detecting depth discontinuities is an important problem that is rarely emphasized in stereo matching. We have presented an algorithm that sacrifices the usual goal of accurate scene depth for crisp discontinuities. The algorithm is fast and able to compute disparities and depth discontinuities in some situations where previous algorithms would fail. Moreover, its results are largely independent of the amount of texture in the image. Two significant limitations that point the way for future research are the algorithm's brittleness and the somewhat ad hoc nature of the postprocessor, which should be replaced by a more principled approach without sacrificing speed.

References

[1] H. H. Baker and T. O. Binford. Depth from edge and intensity based stereo. In *IJCAI*, pp. 631–636, 1981.

[2] P. N. Belhumeur and D. Mumford. A Bayesian treatment of the stereo correspondence problem using half-occluded regions. In *CVPR*, pages 506–512, 1992.

[3] S. Birchfield and C. Tomasi. Depth discontinuities by pixel-to-pixel stereo. Technical Report STAN-CS-TR-96-1573, Stanford University, July 1996.

[4] S. D. Cochran and G. Medioni. 3-D surface description from binocular stereo. *IEEE Trans. on Pattern Analysis and Machine Intell.*, 14(10):981–994, 1992.

[5] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. Maggs. A maximum likelihood stereo algorithm. *Comp. Vision and Image Understanding*, 63(3):542–567, 1996.

[6] P. Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *IJCAI*, pages 1292–1298, 1991.

[7] D. Geiger, B. Ladendorf, and A. Yuille. Occlusions and binocular stereo. *International Journal of Computer Vision*, 14(3):211–226, 1995.

[8] S. S. Intille and A. F. Bobick. Disparity-space images and large occlusion stereo. In *Proc. of the 3rd European Conf. on Comp. Vision*, pages 179–186, 1994.

[9] D. G. Jones and J. Malik. Computational framework for determining stereo correspondence from a set of linear spatial filters. *Image and Vision Computing*, 10(10):699–708, 1992.

[10] J. J. Little and W. E. Gillett. Direct evidence for occlusion in stereo and motion. *Image and Vision Computing*, 8(4):328–340, 1990.

[11] A. Luo and H. Burkhardt. An intensity-based cooperative bidirectional stereo matching with simultaneous detection of discontinuities and occlusions. *Intl. Journal of Computer Vision*, 15(3):171–188, 1995.

[12] Y. Ohta and T. Kanade. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Trans. on PAMI*, 7(2):139–154, 1985.

[13] P.-S. Toh and A. K. Forrest. Occlusion detection in early vision. In *ICCV*, pages 126–132, 1990.

[14] L. E. Wixson. Detecting occluding edges without computing dense correspondence. In *Proceedings of the DARPA Image Understanding Workshop*, 1993.

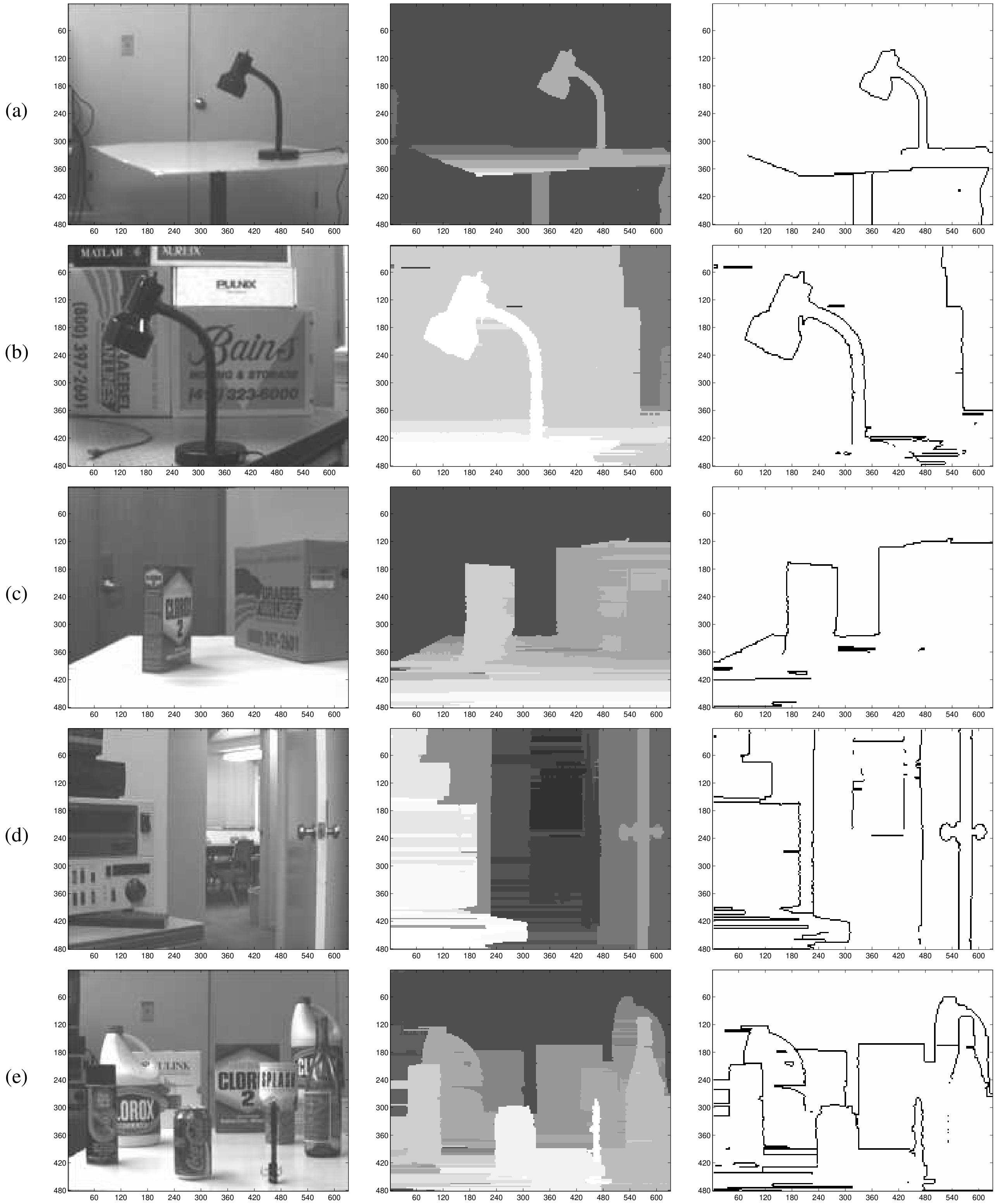

Figure 8: The left image, the disparity map, and the depth discontinuities. These figures are also available from the World Wide Web at `http://vision.stanford.edu`.