Depth maps can be constructed for missing parts which are planar or simply curved surfaces. Missing parts are discovered in a hierarchical fashion, and interpolated using the linear Coons surface model blending function at the bottom of the pyramid.

This paper has also explored the range and intensity edge correspondence issue further than previous researchers. In particular, it was found that the most reliable edge correspondences are achieved when the same edge detection and thinning method is used for all edges. Also, a more general-purpose method of finding edge correspondences has been presented which records the actual region of equivalence.

ACKNOWLEDGMENT

The images used in this paper are from the MSU Pattern Recognition and Image Processing Lab at Michigan State University Laboratory. Special thanks to Patrick Flynn for providing access to the registered intensity images.

REFERENCES

[1] N. Alvertos, B. Dragana, and R. Gonzalez, "Camera geometries for image matching in 3-D machine vision," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 9, pp. 897–915, 1989.
[2] N. Ayache, *Artificial Vision of Mobile Robots: Stereo Vision and Multisensory Perception*. Cambridge, MA: MIT Press, 1991.
[3] P. Besl, "Active optical range imaging," in *Advances in Machine Vision*, J. L. C. Sanz, Ed. Berlin: Springer-Verlag, 1989, pp. 1–63.
[4] P. Burt, "The pyramid as a structure for efficient computation," in *Multiresolution Image Processing and Analysis*, A. Rosenfeld, Ed. Berlin: Springer-Verlag, 1984, pp. 6–35.
[5] W. Chen, W. Lie, and Y. Chen, "Segmented description of 3-D objects with range and intensity data," in *Int. Conf. Image Processing*, 1989, pp. 212–216.
[6] C. C. Chu and J. K. Aggarwal, "Image interpretation using multiple sensing modalities," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, no. 8, pp. 840–847, 1992.
[7] S. Das and N. Ahuja, "Integrating multiresolution image acquisition and coarse-to-fine surface reconstruction from stereo," in *Proc. IEEE Workshop Interpretation 3D Scenes*, 1989, pp. 9–15.
[8] B. Gil, A. Mitchie, and J. Aggarwal, "Experiments in combining intensity and range edge maps," *Comput. Vision Graphics Image Processing*, vol. 21, no. 3, pp. 395–411, 1983.
[9] W. Grimson, "Computational experiments with a feature based stereo algorithm," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 7, no. 1, pp. 17–34, 1985.
[10] W. Hoff and N. Ahuja, "Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 2, pp. 121–136, 1989.
[11] T. Hong, M. Shneier, and A. Rosenfeld, "Border extraction using linked edge pyramids," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-12, no. 5, pp. 660–668, 1982.
[12] T. Hong and M. Shneier, "Extracting compact objects using linked pyramids," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 6, no. 2, pp. 229–237, 1984.
[13] R. C. Jain and A. K. Jain, Ed. *Analysis and Interpretation of Range Images*. Berlin: Springer-Verlag, 1990.
[14] R. Jarvis, "A perspective on range finding techniques for computer vision," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 5, no. 2, pp. 122–139, 1983.
[15] E. P. Krotkov, *Active Computer Vision by Cooperative Focus and Stereo*. Berlin: Springer-Verlag, 1989.
[16] Z. N. Li and G. Hu, "On edge preservation in multiresolution images," *CVGIP: Graphical Models and Image Processing*, vol. 54, no. 6, pp. 461–472, 1992.
[17] D. Marr and T. Poggio, "A computational theory of human stereo vision," *Proc. Royal Soc. London (Series B)*, vol. 204, pp. 301–328, 1979.
[18] J. Mayhew and J. Frisby, "Psychophysical and computational studies towards a theory of human stereopsis," *Arti. Intell.*, vol. 17, pp. 349–385, 1981.
[19] R. Nevatia and K. Babu, "Linear feature extraction and description," *Comput. Graphics Image Processing*, vol. 23, no. 1, pp. 67–91, 1980.
[20] D. Rogers and J. Adams, *Mathematical Elements for Computer Graphics*, 2nd Ed. New York: McGraw-Hill, 1990.
[21] D. Terzopoulos, "Multilevel reconstruction of visual surfaces: Variational principles and finite-element representations," in *Multiresolution Image Processing and Analysis*, A. Rosenfeld, Ed. Berlin: Springer-Verlag, 1984, pp. 237–310.
[22] C. Thorpe, M. H. Hebert, T. Kanade, and S. A. Shafer, "Vision and navigation for the Carnegie-Mellon Navlab. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10, no. 3, pp. 362–373, 1988.
[23] L. Uhr, "Parallel, hierarchical software/hardware pyramid architectures," in *Pyramid Systems for Computer Vision*, V. Cantoni and S. Levialdi, Eds. Berlin: Springer-Verlag, 1986, pp. 1–20.
[24] N. Yokoya and M. Levine, "Range image segmentation based on differential geometry," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 6, pp. 643–649, 1989.

Stereo Correspondence Based on Line Matching in Hough Space Using Dynamic Programming

Ze-Nian Li

***Abstract*—This paper presents a method of using Hough space for solving the correspondence problem in stereo vision. It is shown that the line-matching problem in image space can readily be converted into a point-matching problem in Hough (ρ–θ) space. Dynamic programming can be used for searching the optimal matching, now in Hough space. The combination of multiple constraints, especially the natural embedding of the constraint of figural continuity, ensures the accuracy of the matching. The time complexity for searching in dynamic programming is $O(pmn)$, where m and n are the numbers of the lines for each θ in the pair of stereo images, respectively, and p is the number of all possible line orientations. Since m and n are usually fairly small, the matching process is very efficient. Experimental results from both binocular and trinocular matchings are presented and analyzed.**

I. INTRODUCTION

The recovery of depth information is important for 3-D image analysis. One method for depth recovery is *stereo vision*, in which pairs of images from horizontally and/or vertically displaced cameras are used. One of the most difficult problems in stereo vision is *correspondence* [1]. Once corresponding points in the pair of images are identified, their *disparity* values can be calculated and used to recover the depth.

We are developing a vision system for mobile robots. The proposed domain is an office environment. The robot is planned to walk in corridors and rooms to fetch and deliver simple objects. Both stereo and laser range data will be used. Since most of the objects of interest will be man-made, it is natural to think of straight lines as the main feature for stereo matching. The stereo algorithm described in this paper, can quickly render a nondense depth map

Manuscript received August 22, 1991; revised March 5, 1993. This work was supported in part by the National Sciences and Engineering Research Council of Canada under grant OGP-36726 and a research grant from the Centre for Systems Science of Simon Fraser University.

Z. N. Li is with the School of Computing Science, Simon Fraser University, Burnaby, B.C. V5A 1S6, Canada.

IEEE Log Number 9212949.

which will help the robot to overcome most parts of the corridors and offices. A range-guided stereo system, as described in another paper [2], will be invoked for generating a dense depth map when close-up views are necessary.

Baker and Binford [3], Ohta *et al.* [4], [5] presented interesting stereo-matching algorithms using *dynamic programming*. In Ohta and Kanade's paper [4], edge-delimited intervals were used as elements to be matched. The correspondence problem was cast as a problem of finding an optimal path in search spaces. In order to incorporate the *constraint of figural continuity* [6], a 3-D search space was used. It was, therefore, relatively computationally complex. The later paper by Ohta *et al.* [5], employed "collinear trinocular stereo" to cope with the occlusion problem. The issue of figural continuity was, however, not addressed.

During the past decade, more and more feature-based stereo-matching algorithms have been developed, with many using linear features [7], [8]. Several state-of-the-art robot systems primarily used line features for vision-guided navigation and object recognition, especially in man-made environment [9], [10]. One of the major advantages of the line-matching techniques is that they naturally incorporate the constraint of figural continuity. The process for matching linear features can still be quite complex. In general, the performance and speed of the existing stereo systems are yet to be satisfactory.

The Hough transform [11] is known to be suitable for extracting lines and curves. In this process, lines (curves) in image space are transformed into peak points in the parameter space, or *Hough space*. We discovered that the line-matching problem in the image space can be readily converted into a peak point-matching problem in Hough space. Dynamic programming can be performed in Hough space, namely, the matching entities are the peak points in Hough space. It turns out that if the lines are represented by parameters ρ and θ, then the search process for the dynamic programming can be independently executed for each θ, and is, therefore, very efficient. After a correct matching between corresponding lines is made, it is not difficult to obtain the disparity values, with a refined higher accuracy, for the individual corresponding points on the pairs of matched lines. Compared with the original dynamic programming approach by Ohta and Kanade [4], the constraint of figural continuity is embedded in our line-based approach, and hence, only one search in the Hough space is needed. The complexity of the search process is thereby significantly reduced.

As shown by Burt and Julesz [12], binocular fusion cannot be obtained when the *disparity gradient* exceeds a certain limit. The disparity gradient limit implies the *constraint of nonreversal order*, i.e., the left-to-right order of lines cannot be reversed in one eye with respect to the other. This constraint introduces only a minor (and usually acceptable) restriction on the real applications. In this paper, a peak ordering algorithm based on the constraint of nonreversal order, is presented to ensure the optimality of the dynamic programming when the orientations of the corresponding lines in the stereo images differ.

It is known that if the two cameras of a binocular system are displaced horizontally, then the horizontal lines in two images cannot be uniquely matched unless the end points of the horizontal lines can be located. A good solution to this is the use of three cameras as suggested by [13]–[15]. The trinocular stereo vision facilitates an additional powerful constraint, the *trinocular uniqueness constraint* [13], i.e., the horizontal and vertical disparities are identical, provided the camera displacements in both directions are equal. It will be effectively accommodated in the search process of our dynamic programming.

The proposed Hough method for line matching employs multiple constraints (epipolar, uniqueness, disparity gradient, figural continuity, and trinocular uniqueness constraints). Especially, it has the merits of naturally enforcing the constraint of figural continuity and speeding up the matching process. Moreover, it could be used to simultaneously yield certain boundary descriptions in Hough space which can be very useful for surface interpolation as suggested, for instance, by Hoff and Ahuja [16] in their integrated approach for Surface from Stereo.

The organization of the paper is as follows: Section II describes our algorithm of line matching in Hough space. Section III discusses dynamic programming for line matching in Hough space. Section IV presents experimental results. Section V concludes the paper.

II. Stereo Correspondence in Hough Space

This section will describe the method for line matching in Hough space. Since most procedures for binocular matching can be shared by trinocular matching in an obvious way, most of the descriptions in this paper will be based on binocular stereo vision. They can readily be generalized to trinocular matching, with the aid of the discussions in the last part of Section III-C.

Let L_l and L_r be the corresponding lines in the left and right images, represented by

$$L_l\colon \rho_l = x_l \cos\theta_l + y_l \sin\theta_l,$$

$$L_r\colon \rho_r = x_r \cos\theta_r + y_r \sin\theta_r,$$

where ρ_l and ρ_r denote the normal distances from the lines to the origins, and θ_l and θ_r are the edge gradient directions which are perpendicular to the line orientations. For stereo images, two (ρ–θ) accumulator arrays can be used for the left and right images, respectively. After the points on L_l and L_r vote on the accumulators, peaks will be formed at (ρ_l, θ_l) in the left Hough space, and at (ρ_r, θ_r) in the right Hough space, each representing a line. Because of the epipolar constraint, each pair of matching points on the corresponding lines in the left and right images are considered to have an identical y coordinate, i.e., $y_l = y_r$. Moreover, if the displacement b of the two cameras is relatively small and the objects are relatively far away from the cameras, then it can be assumed that $\theta_l \approx \theta_r$. Accordingly,

$$(\rho_l - \rho_r) \approx (x_l - x_r)\cos\theta, \tag{1}$$

where θ is approximately equal to θ_l and θ_r.

With the above assumption, the following algorithm can be used for deriving disparity values.

Algorithm (Stereo Correspondence Based on Line Matching)

1) For each peak in Hough space, maintain a list of the coordinates for the points that voted for this peak.
2) *Find the corresponding peaks representing L_l and L_r in the left and right Hough spaces.*
   The search space for matching peaks in the Hough spaces in quite limited, because:
   a) For each peak with θ_l in the left accumulator, only peaks with $\theta_l \pm \Delta\theta$ are possible matching candidates in the right accumulator,[1] while $\Delta\theta$ is a small number.
   b) According to (2.1), if the estimated disparity ($d = x_l - x_r$) values are in the range of $[d_1, d_2]$, where d_1 and d_2 are the lower and upper bounds of the possible disparities,

[1]In this paper we only discuss matching from the left image to the right. Although matching from right to left can be similarly conducted and both results must then be combined in a cooperative way.

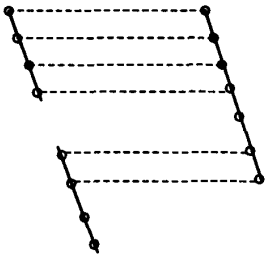


Fig. 1. Assign disparity values after line matching.

then the displacement $(\rho_l - \rho_r)$ will be in the range of $[d_1 \cos\theta, d_2 \cos\theta]$. Thus, only peaks with $\rho_r \in [\rho_l - d_1 \cos\theta, \rho_l - d_2 \cos\theta]$ are possible matching candidates in the right accumulator array.

3) *Assign disparity values to individual points on the matched lines L_l and L_r.*
Points that are on the same epipolar line are now readily put in pair. Their $x_l - x_r$ are taken as disparity (d) values. Fig. 1 illustrates a possible point-matching situation after the two lines have been chosen as corresponding pairs. One of the advantages of using the Hough transform is that it is tolerant to noisy and broken lines. In case a slightly broken line is caused by a partial occlusion in one image but not in the other, two corresponding peaks will still be formed and matched. As shown in Fig. 1, a reasonable match between most points is still possible.[2]

In most cases, the actual disparity d, assigned to each individual point, is slightly different from the estimated disparity for the whole line, since a small nonzero $\Delta\theta$ is allowed. In other words, the disparity for each individual point is refined to a higher accuracy.

III. Dynamic Programming for Line Matching in Hough Space

This section shows the most important part, Step 2 of the proposed algorithm, i.e., dynamic programming in Hough space. To start the description, it is temporarily assumed that $\theta_l = \theta_r$, i.e., $\Delta\theta = 0$. This assumption will be relaxed in Section III-B.

As depicted in Fig. 2(a), for a particular θ there may exist more than one contending line in both left and right images. Accordingly, more than one peak is located for the same θ in both left and right Hough spaces. In Fig. 2(a), the corresponding lines are drawn as thicker ones. Often, one-to-one correspondence does not exist between lines in the left and right images. This is because 1) the leftmost lines in the left image may be shifted out of the image plane in the right image, and vice versa; 2) the occlusions of lines may be different from the left and the right views; 3) the noise and reflectance in the two images may differ. In general, for each θ, it is necessary to find the correspondence between n peak points in the left Hough space and m peak points in the right Hough space.

For dynamic programming in Hough space, an $m \times n$ search plane is constructed for each possible θ value. Fig. 2(b) illustrates such a search plane. The horizontal axis shows the peaks in the left Hough space and the vertical axis shows the peaks in the right Hough space. Along each θ in the Hough space, peaks are scanned from left to right, namely, peaks are ordered according to their ρ values. In this example, $n = 4$ and $m = 5$.

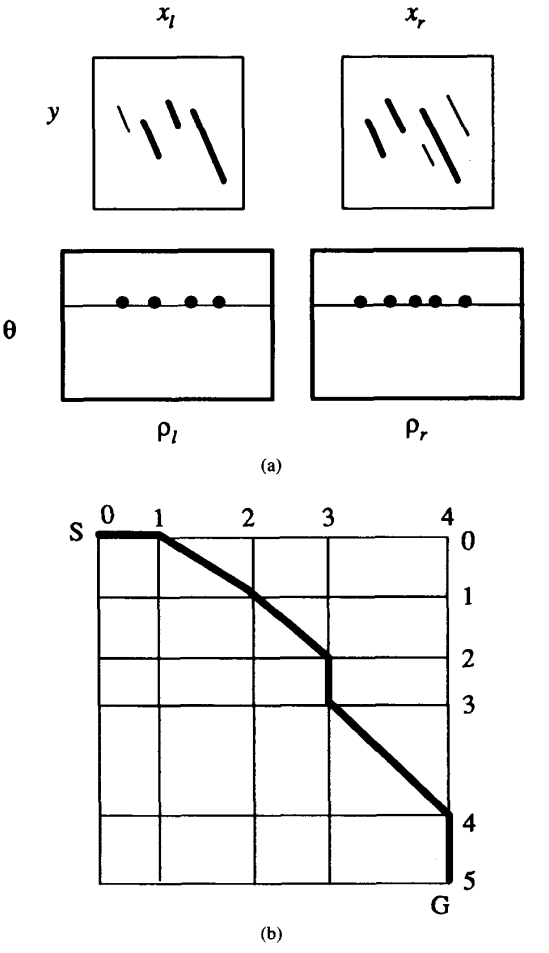


Fig. 2. Dynamic programming for line matching in Hough space. (a) Lines in image $(x-y)$ space and their corresponding peak points in Hough $(\rho-\theta)$ space. (b) The search plane for matching lines in (a).

For line matching in Hough space, if $\theta_l = \theta_r$ is presumed, i.e., all contending m lines in the right image and n lines in the left image are parallel, then the ρ values of the peaks can be used to determine the partial order of the peaks in the Hough spaces. It is because:

1) For each direction θ_0 the peaks are all in 1-D ρ–θ_0 subspaces.
2) The order of the peaks in each 1-D ρ–θ_0 subspace has a fixed and consistent relationship with the order of the lines in each image.

In this case, the constraint of nonreversal order also applies to the corresponding peaks in the Hough subspaces along the ρ dimension.

A. Optimal Solution Path

The process of dynamic programming in the Hough space is similar to the dynamic programming in image space described by Ohta and Kanade [4]. Instead of matching intervals in image space, peaks in Hough space which represent lines in the image space are

[2]The term "line matching" has been loosely used in this paper. Strictly speaking, "collinear points" are being matched. Peaks in the Hough spaces represent the collinear points (whether they are connected or not). Optimal matches between the corresponding peaks are obtained in the Hough space using dynamic programming. Afterwards, disparity values are assigned to the individual points on each epipolar line.

matched. The correspondence problem is now a problem of finding an *optimal solution path* (with the least cost) on the search plane, where nodes in the search plane correspond to potential matches of peaks from the Hough space, and a solution path is any path from S to G [from (0, 0) to (m, n)].

Each of the three types of segments below is a *primitive path* and is assigned a cost. Because of the nonreversal ordering constraint, starting from S, a path can be extended towards only one of the three directions: east, south, or southeast.

- The horizontal segment from $(i, j-1)$ to (i, j) on a path, corresponds to no match for line j in the left image, e.g., the dark horizontal segment from (0, 0) to (0, 1) in Fig. 2(b) indicates that the first (leftmost) line in the left image has no matching line in the right image.
- The vertical segment from $(i-1, j)$ to (i, j) on a path, corresponds to no match for line i in the right image.
- The diagonal segment from $(i-1, j-1)$ to (i, j) on a path, corresponds to a match between line j in the left image and line i in the right image, e.g., the first dark diagonal segment from (0, 1) to (1, 2) in Fig. 2(b) represents the match between the second leftmost line in the left image and the first (leftmost) line in the right image.

Let $C(t)$ be the cost of the optimal path from the origin (0, 0) to node t, and $c(t, t-i)$ be the cost of the primitive path from $t-i$ to t. Then the local optimal cost at any node in the search plane can be recursively defined as:

$$C(S) = 0,$$

$$C(t) = \min_{\{i\}} \{c(t, t-i) + C(t-i)\},$$

where

$$S = (0, 0); \quad i = (i, j), \quad 0 \le i \le 1, 0 \le j \le 1,$$

$$\text{and } i + j \neq 0.$$

The path that renders $C(G)$ at node $G = (m, n)$ is the optimal solution path.

It is presumed that the search process is Markovian [4], namely, $C(t)$ only depends on its immediate predecessor's $C(t-i)$ and the $c(t, t-i)$. It does not depend on the previous history, i.e., the way of deriving $C(t-i)$.

B. Peak Ordering

The optimality of the dynamic programming relies on the consistent ordering of the peaks in both Hough spaces. If the orders of the peaks for matching lines L_{l1}–L_{r1}, and L_{l2}–L_{r2} are not the same in the two Hough spaces, then at least one of these lines will simply be overlooked by the matching process.

In general, when a line in the 3-D scene is not parallel to the image planes, its projections onto the left and right images are not in parallel. In other words, $\theta_r = \theta_l + \Delta\theta$ and the assumption that $\Delta\theta \approx 0$ is now relaxed. This subsection examines the peak-ordering problem when relatively large $\Delta\theta$ values are encountered. It should be reemphasized that the presumption that lines will never have reversed orders in the left and right images has always been made throughout this paper.

1) Problem of Peak-Ordering in 2-D Hough Subspaces: As stated above, for each line orientation with a gradient direction θ_0, when $\Delta\theta = 0$ was assumed, the ρ values of the peaks could be used to determine the partial order of the peaks in the 1-D Hough subspaces (where $\theta = \theta_0$). If $\Delta\theta \neq 0$, the peaks will reside in a 2-D ρ–θ subspace, where $\theta \in [\theta_0 - \Delta\theta, \theta_0 + \Delta\theta]$, and their ordering becomes a nontrivial problem. Unfortunately, the ρ values can no longer be directly used to determine the partial order of the peaks in the Hough subspace. Fig. 3 illustrates this problem. Although lines L_1 and L_2 have the same order in their respective left and right image spaces (while scanning from left to right on each epipolar line, L_1 precedes L_2), we have $\rho_{l1} < \rho_{l2}$, whereas $\rho_{r1} > \rho_{r2}$.

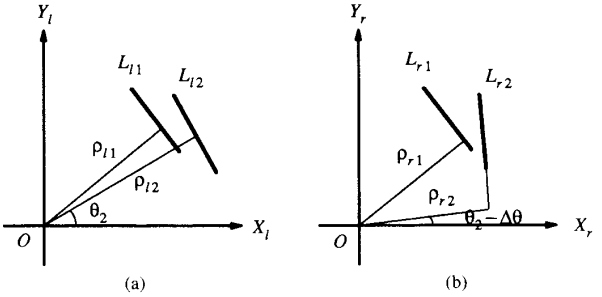


Fig. 3. An example of reversed ρ-order. (a) Left image. (b) Right image.

2) A Peak-Ordering Algorithm: This subsection describes an insert-sort algorithm which is needed for peak ordering in a 2-D (ρ–θ) Hough subspace, where $\theta \in [\theta_0 - \Delta\theta, \theta_0 + \Delta\theta]$. Since peaks in the Hough space and lines in the image have a one-to-one mapping, the peak ordering will actually be accomplished by examining the order of the lines in the image. The correctness of this algorithm is based on the fact that the order of the lines in both stereo images is nonreversal.

Let (x_{k1}, y_{k1}) and (x_{k2}, y_{k2}) be two endpoints of line L_k, and $y_{k1} < y_{k2}$.

Algorithm (Peak Ordering):

1) Sort all the lines with $\theta \in [\theta_0 - \Delta\theta, \theta_0 + \Delta\theta]$ in increasing order of the y-coordinates (y_{k1}) of their lower endpoints, and store them in INITIAL_LIST.
2) Repeat until INITIAL_LIST is empty:
   Fetch the first line L in INITIAL_LIST. If RESULT_LIST is empty, insert L in RESULT_LIST; otherwise, compare L with each line L_k in RESULT_LIST:
   if L is *ahead_of* L_k, insert L before L_k;
   otherwise (L is not *ahead_of* any L_k), append L at the end of RESULT_LIST.

- To ensure that the algorithm works properly, cases of intersecting lines are not allowed. If two lines intersect, one of them must be split into two at the intersecting point.
- The function *ahead_of* simply checks the geometrical relationships between two lines. For nonintersecting lines L_k and L in step 2), let y_{k1}, y_{k2}, and y_1, y_2 be their y-coordinates of respective endpoints; then from the above discussion we have $y_{k1} < y_{k2}$, $y_1 < y_2$, and $y_{k1} < y_1$.

```
function ahead_of;
  begin
    if y1 > yk2/*Lk and L do not overlap in Y dimension
        (nonoverlapping Y-extents)*/
      then return nil;
    else if the endpoint y1 is at the left of line Lk
          then return true;
        else return nil;
  end;
```

C. Cost Functions

The cost c for each primitive path on the search plane is assigned by heuristic *cost functions*. The following are used in this paper. The first function (c_1) is applicable to all primitive paths, while the

second and third (c_2 and c_3) are only applied to the diagonal primitive paths. When multiple cost functions are used, a weighted sum of costs $c = \Sigma_i w_i c_i$ is taken, where w_i are adjustable weights.

1) Maximize the Number of Points Matched (c_1): The heuristic is that, usually, less points can be put in pairs, if erroneous correspondence is made among the contending lines. Since the horizontal and vertical segments on a path in the search plane imply no matches, their cost is set to be equal to the length of the line being skipped. This could be costly—in other words, the skipping of long lines is discouraged. The diagonal segment represents a match between a pair of lines. As shown in Fig. 1, there might not be a perfect point-to-point match between the designated matching lines. Let l_1 and l_2 be the lengths of the lines L_l and L_r, and h be the number of the points matched on each line. Then the cost assigned to the diagonal segment in the search plane is

$$c_1 = l_1 + l_2 - 2h. \tag{2}$$

Fig. 4 shows a small example of an optimal path with a cost function that maximizes the number of points matched between six left lines and six right lines. These lines all have approximately the same θ. The lengths for the six left lines are 3, 11, 13, 18, 13, 8. The lengths for the right lines are 12, 14, 21, 12, 8, 7. The number in $(0, j)$ indicates the cost for skipping the first j left lines. The number in $(i, 0)$ indicates the cost for skipping the first i right lines. In general, all the numbers indicate the optimal cost C for getting to the node. Let us take node (4, 5) as an example of the calculation of $C(t)$. As said before, there are three paths to reach the node (4, 5) and their cost are:

$$c((4, 5), (4, 4)) + C(4, 4) = 13 + 22 = 35,$$
$$c((4, 5), (3, 5)) + C(3, 5) = 12 + 23 = 35,$$
$$c((4, 5), (3, 4)) + C(3, 4) = 5 + 10 = 15.$$

Obviously, the last has the least cost of 15.[3] Therefore, $C(4, 5) = 15$.

2) Compare the Local Property at the Vicinity of the Lines (c_2): Edge magnitudes along a line can be used to measure the difference between the gray-level intensities at the two sides of the line. It can be enforced that two correctly matched edge lines should have similar edge magnitudes. A higher cost can be assigned to the candidate pair that has larger differences in their local properties. Let h be the number of possible pairs of matching points on lines L_l and L_r, and S_l^k and S_r^k be the edge magnitudes at the kth point. The cost function for measuring the difference of the local properties for L_l and L_r is defined as

$$c_2 = \frac{1}{h} \sum_{k=1}^{h} |S_l^k - S_r^k|. \tag{3}$$

3) Enforce the Trinocular Uniqueness Constraint (c_3): The trinocular uniqueness constraint is effective in reducing false matches [13]. It can be incorporated into a heuristic function in the following way:

Assume the left and right cameras are arranged as before; the third camera is placed above the left camera with the same displacement b. The image obtained from the left camera will now be called *base* (B) image, and the other two images *horizontal* (H) and *vertical* (V) *images*, respectively [15]. Two dynamic programming search spaces will be created, one for the horizontal B–H matching and the other for the vertical V–B matching. To illustrate

[3] $c((4, 5), (3, 4))$ reflects the cost of the partial match between the fifth left line and the fourth right line. Ten points from each line are matched. The cost for this diagonal segment is $l_1 + l_2 - 2h = 13 + 12 - 2 \times 10 = 5$.

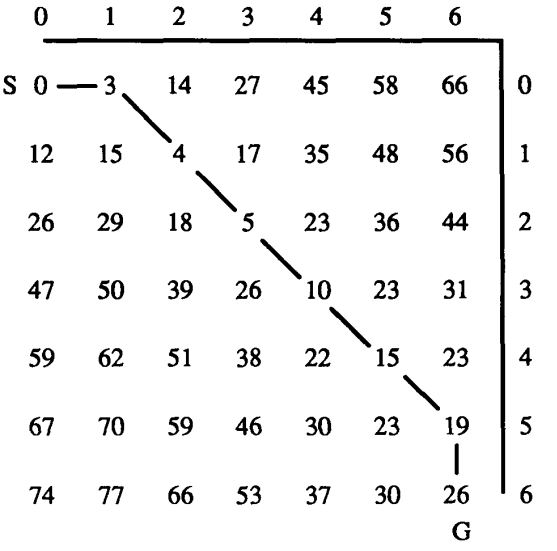


Fig. 4. An optimal path that maximizes the number of points matched.

graphically, they are drawn together in Fig. 5 with the rotated V–B search space placed on the top of the B–H search space, both sharing the middle axis for the base image. As shown, the optimal solution path for the B–H matching runs from S to G_h, whereas the path for the V–B matching runs from S to G_v.

The trinocular uniqueness constraint implies $\overline{d_h} = \overline{d_v}$, where $\overline{d_h}$ and $\overline{d_v}$ are the average horizontal and vertical disparity values along the correctly matched lines. Let $\overline{d_h}(i, j)$ be the average disparity value associated with the diagonal segment $(i - 1, j - 1)$ to (i, j) in the B–H search space, the cost function for enforcing the trinocular uniqueness constraint at node (i, j) is defined as:

$$c_3(i, j) = \Delta d^2(i, j), \tag{4}$$

where

$$\Delta d(i, j) = \operatorname*{Min}_{k=k_1}^{k_2} |\overline{d_h}(i, j) - \overline{d_v}(k, j)|. \tag{5}$$

$\Delta d(i, j)$ measures the minimum difference between $\overline{d_h}(i, j)$ and $\overline{d_v}(k, j)$ of the potential matching lines within the range $[k_1, k_2]$ in the V–B search space. As shown in Fig. 5, (k_0, j) is the node in the V–B search space that offers the minimum difference Δd.

D. Time Complexity for Searching in Dynamic Programming

Since each node t has only three possible predecessors $t - i$, i.e., its immediately west, north, or northwest neighboring node, the computation of $C(t)$ for all the nodes can be done in one scan (row-by-row) through the 2-D search plane, starting from S and ending at G.

Taking into account of the fact that $\Delta\theta \neq 0$, the number of nodes in the search plane for each θ is $O(mn)$, where m and n are the numbers of the lines for each θ in the pair of stereo images, respectively. At each node only three primitive paths are examined, denoted as $u = 3$. It is also suggested in Step 2 of the algorithm that only peaks with $\rho_r \in [\rho_l - d_1 \cos\theta, \rho_l - d_2 \cos\theta]$ are possible matching candidates in the right accumulator array. That means only a portion k of the nodes are to be searched ($k < 1$). Altogether, the number of primitive paths to be examined is in the order of $k \times m \times n \times u$. It follows that the computational cost for each θ is $O(mn)$. Since the search will be performed for all possible line

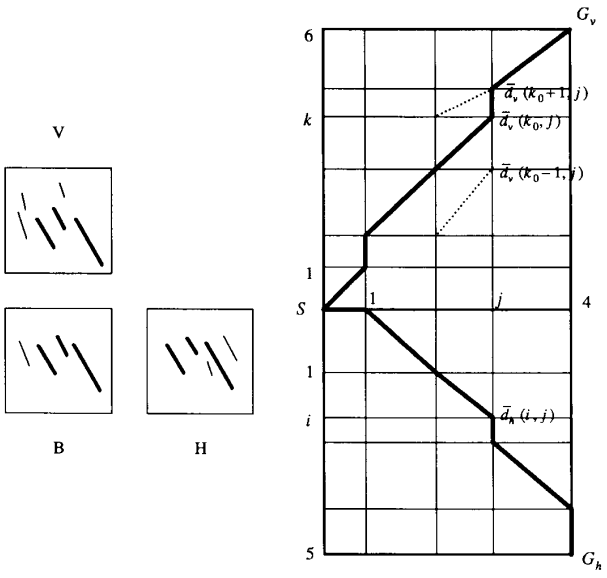


Fig. 5. Search spaces for trinocular matching.

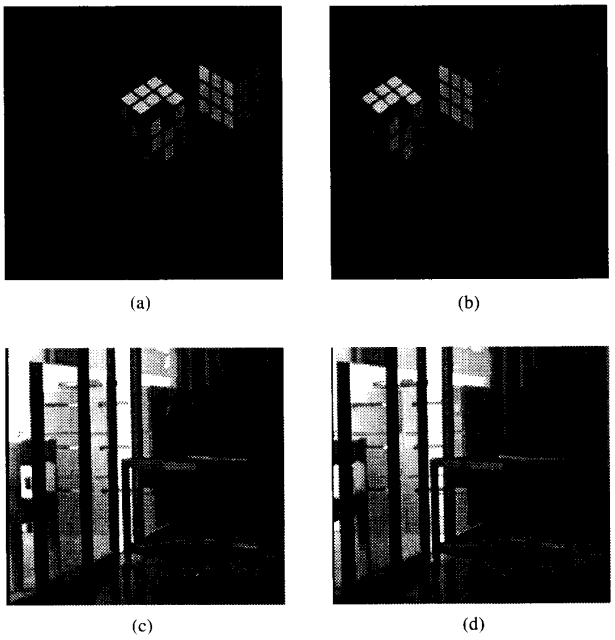


Fig. 6. Test images for the binocular scenes. (a) Left cube. (b) Right cube. (c) Left corridor. (d) Right corridor.

orientations, the overall time complexity is $O(pmn)$, where p is the number of all possible θ's. There are, for example, 36 possibly different θ's in the Hough space, and m and n are usually fairly small. This makes the search very fast.

The enforcement of the trinocular uniqueness constraint has its overhead. While performing horizontal matching, the disparities of potential vertical matches must be examined, and vice versa. However, only a constant number of the potential matches within a certain disparity range are examined each time. Hence, the time complexity remains $O(pmn)$.

IV. Experimental Results

The proposed method was tested using real-world images. The first two sets of our test data are binocular stereo images in which two cameras are mounted on the same horizontal level and separated with a displacement $b = 4$ in. The first set has two Rubik cubes in each image [Fig. 6(a), (b)]. They are frequently used in the stereo-vision literature, because the repetitive pattern often causes some difficulty for the correspondence process. The second stereo pair is an ordinary indoor scene (Fig. 6(c), (d)). It is a corridor scene, typical for an office environment. The first room is an office with sliding glass doors and windows. A cart is placed in the corridor to add a little more complexity to the scene. The third set is a trinocular office scene (Fig. 7(a)–(c)). The horizontal and vertical camera displacements are both $b = 4$ in.

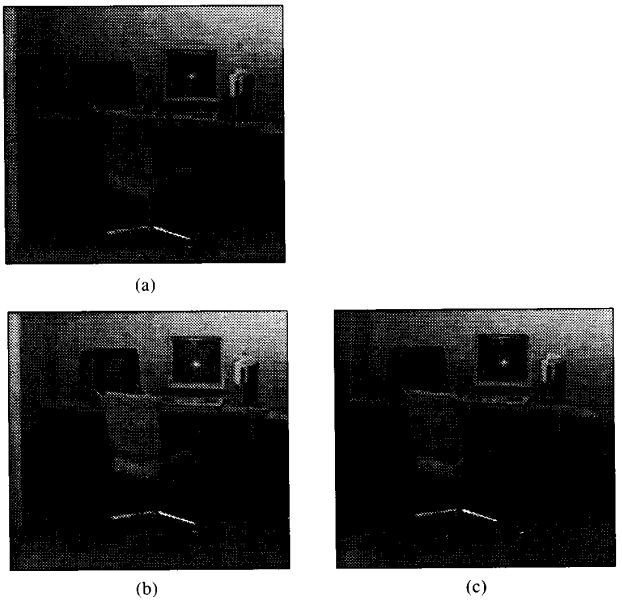
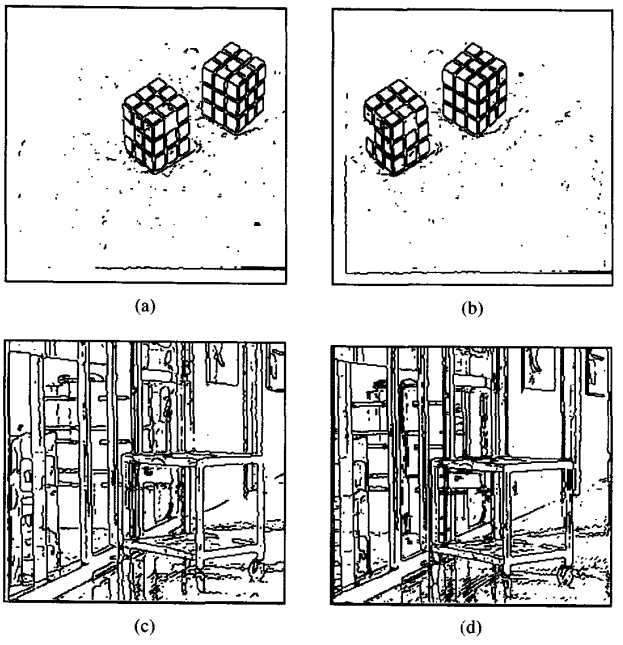


Fig. 7. Test images for the trinocular "office" scene. (a) Vertical. (b) Base. (c) Horizontal.

A. Edge Detection

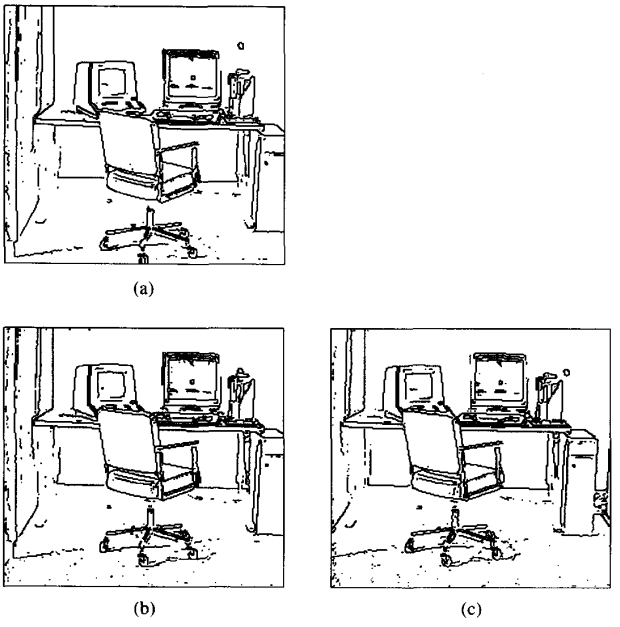
A gradient-based edge operator adopted by Rosenfeld, Ornelas, and Hung [17] is used. Basically, two Sobel gradient masks are used to derive approximate partial derivatives along the horizontal and vertical directions. They are then combined to yield the gradient (magnitude and direction) information. The edges thus extracted are usually more than one pixel wide. The *nonmaximum-suppression* technique [17] is used to reduce their edge widths to one. A pixel P will survive as an edge pixel, if its edge magnitude is greater than the edge magnitudes of both neighbors that are along the edge direction and have similar edge directions as pixel P. Figs. 8 and 9 are the edge maps after pyramid linking for the test images.

The recursive edge detector by Deriche [18] was also tried for comparison. Its edge detection result was quite similar to the Sobel result. The Sobel operator was chosen because of its readiness for parallel processing.



(a) (b) (c) (d)

Fig. 8. Edge maps for the binocular scenes. (a) Left cube. (b) Right cube. (c) Left corridor. (d) Right corridor.



(a) (b) (c)

Fig. 9. Edge maps for the trinocular "office" scene. (a) Vertical. (b) Base. (c) Horizontal.

B. Implementing the Hough Transform

As suggested by Duda and Hart [11], a ρ–θ accumulator array is used as the Hough space for the lines. The range of the accumulator array is $0° \leq \theta < 360°$, $-\rho_{max} \leq \rho \leq \rho_{max}$. The divisions are every 10° for θ and every 1 pixel for ρ. Since the gradient information is used, an edge point P votes for only one corresponding point (ρ_0, θ_0) in the accumulator array. After all the edge points in the image space have voted, local maximum points (peaks) are extracted to represent the lines.

Because of the inevitable error of discrete sampling and especially the inaccuracy in the computation of the edge direction θ, a peak in the Hough space for a line is usually not as sharp as a single spike. It is more likely that all the collinear points in the image will map into a cluster in the accumulator array centered at the location of the peak point. We developed a hierarchical peak compaction method [19] which adapts to the error scale at each level and calculates the best achievable accuracy, and, therefore, generates sharp peaks in the Hough space. As an example, Fig. 10 illustrates the Hough accumulator arrays generated by the hierarchical peak compaction method for the cube images. The height of the bars represents the total votes collected for each peak.

As described in Step 1 of the algorithm in Section II, it is necessary to keep the coordinate information for all the voting edges. Hence, a linked list of edge points is maintained at each peak in the accumulator array. For expedient access, the list is sorted by y and then x values of the edge points. The sorting turns out to have no overhead, since the edge points were examined in order, at the time when their votes were taken.

C. Dynamic Programming for Matching

The peak ordering and searching process for the dynamic programming have been discussed in detail in Section III. In our implementation, relatively large $\Delta\theta$ is allowed. For the binocular images, the matching is performed from the left to right. For the peak points with $\theta_0 - 5° \leq \theta \leq \theta_0 + 5°$ in the left Hough space, potential matches with peak points in a subspace $\theta_0 - 15° \leq \theta \leq \theta_0 + 15°$ in the right Hough space are considered. In the trinocular case, both base-horizontal and base-vertical matching are performed adopting the same range of $\Delta\theta$.

As an example, Fig. 11 shows the search space of the dynamic programming for the left and right cube images when θ_0 130°. Twelve of the slant lines have their edge gradient directions falling into this range in both images. In addition, six and four noisy and short line segments are also detected by the Hough transform from the left and right images, respectively. Hence, the size of the search space is $m = 16$ and $n = 18$. For the cost function C, a combination of two criteria, applicable to binocular matching is used. More weight has been placed on the first criterion, i.e., maximizing the number of points that can be matched ($w_1 = 0.8$ and $w_2 = 0.2$). The search result is indicated by the optimal solution path in Fig. 11 in which the twelve diagonal line segments correspond to all correct matches. All the short lines (noise) are skipped in this case, although there could sometimes be incidental matches between them as well.

Because each original scene consists of two cubes at different depth and their slant lines have similar θ and ρ values, the peak ordering process turns out to be crucial for generating the correct result. It happens that the peak-ordering problem as illustrated in Fig. 3 occurs quite frequently. Consistent ordering is obtained by the insert-sort peak-ordering algorithm. Intermixed with the noisy short lines, the six slant lines of the lower left cube are placed in front of the six slant lines of the upper right cube in both images. Among the six lines in each group, they are ordered from left to right, in the same way as they appear in the original images.

The trinocular matching shares most procedures with the binocular matching. For vertical (or near vertical) lines, matching only takes place in the base-horizontal pair of images, whereas, for horizontal (or near horizontal) lines, matching only takes places in the

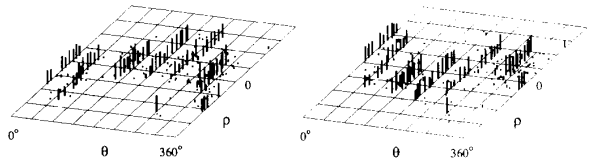


Fig. 10. Hough accumulator arrays for cube images. (a) Left cube. (b) Right cube.

S 0 2 4 6 41 83 112 136 171 206 208 210 212 247 283 310 337 368 397
4—6—8—10 45 87 108 132 167 202 204 206 208 243 279 306 333 364 393
37 39 41 43 16 58 87 111 146 181 183 185 187 222 258 285 312 343 372
78 80 82 84 57 19 48 72 107 142 144 146 148 183 219 246 273 304 333
104 106 108 110 83 45 28 52 87 122 124 126 128 163 199 226 253 284 313
125 127 129 131 104 66 49 33 68 103 105 107 109 144 180 207 234 265 294
128 130 132 134 107 69 52 36 71 106 108 110 112 147 183 210 237 268 297
165 167 169 171 144 106 89 73 40 75 77 79 81 116 152 179 206 237 266
204 206 204 206 183 145 128 112 79 54 56 58 60 95 131 158 185 216 245
206 208 206 208 185 147 130 114 81 56 58 60 62 97 133 160 187 218 247
209 211 209 211 188 150 133 117 84 59—61—63—65 100 136 163 190 221 250
243 245 243 245 222 184 167 151 118 93 95 97 99 78 114 141 168 199 228
278 280 278 280 257 219 202 186 153 128 130 132 134 113 87 114 141 172 201
310 312 310 312 289 251 234 218 185 160 162 164 166 145 119 94 121 152 181
342 344 342 344 321 283 266 250 217 192 194 196 198 177 151 126 103 134 163
374 376 374 376 353 315 298 282 249 224 226 228 230 209 183 158 135 116 145
408 410 408 410 387 349 332 316 283 258 260 262 264 243 217 192 169 150 125
G

Fig. 11. Dynamic programming search space for the cube images when $\theta_0 = 130°$.

base-vertical pair of images. The trinocular uniqueness constraint is only applicable to the slant lines, e.g., $20° \leq \theta \leq 70°$, where matchings could occur in both directions. The weights for computing the cost C of these slant lines are ($w_1 = 0.4$, $w_2 = 0.2$, and $w_3 = 0.4$).

Fig. 12 shows the disparity maps generated by our program. For displaying in black and white, disparities are shown by various gray-level intensities. The points with larger disparities are shown brighter. The binocular configuration is not suitable for stereo-matching between horizontal lines. Hence, matching is not attempted for these lines. The gray outputs for horizontal (or nearly horizontal) lines in Fig. 12(b) are not indications of their disparities. Rather, they just indicate the existence of these edges for ease of viewing. Fig. 12(c), however, is a complete disparity map generated by the trinocular matching.

D. Performance Analysis

Disparity error rate ϵ is used to measure the quality of the stereo matching. ϵ is calculated by carefully examining the disparity map. If a matched point has a disparity deviated from the correct value by more than one, it is counted as an error.

As shown in Table I, $\epsilon = 2.57\%$ for the cube scene, 4.80% for the corridor scene, and 2.98% for the office scene. The above disparity errors compare very well with other reported results [13]–[15], [20]. For the cube images, all 72 pairs of lines are correctly matched, and the disparity errors occur mostly on the slanted edges in the cubes. For the corridor scene, the matching between horizontal lines are not attempted; most disparity errors are again caused by the slant lines and the mismatches on the noisy (textured) floor points in the corridors. The trinocular matching for the office scene indeed, produces good matches for lines of all orientations. Moreover, the error rate $\epsilon = 2.98\%$ is significantly smaller than the $\epsilon = 4.80\%$ in the binocular corridor scene with comparable complexities.

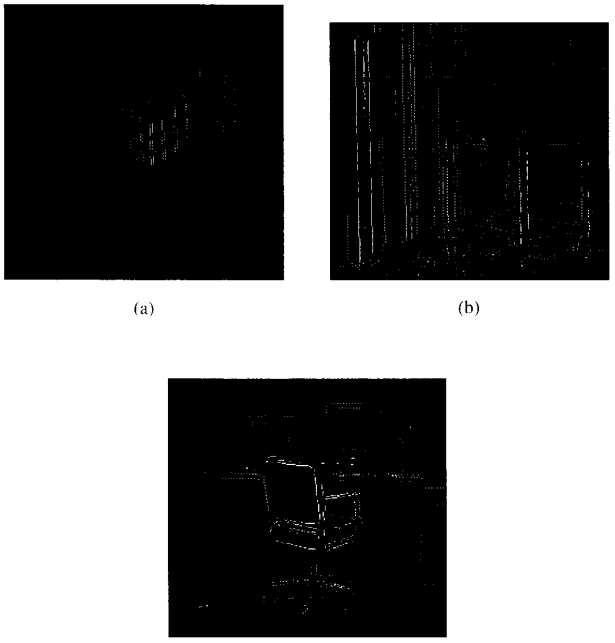


Fig. 12. Disparity maps. (a) Cube. (b) Corridor. (c) Office.

TABLE I
DISPARITY ERROR RATES OF BINOCULAR AND TRINOCULAR METHODS

	Binocular		Trinocular
Images	Cube	Corridor	Office
Disparity error rate (ϵ)	2.57%	4.80%	2.98%

Currently, the entire matching process for the trinocular office scene takes 7.4 sec of cpu time on a Sun SPARC-1: approximately 0.51 sec (7% of the 7.4 sec) for peak ordering, 1.28 sec (17%) for the search process of dynamic programming, 3.6 sec (48%) for the calculation of cost functions and other programming overheads, and 2.0 sec (26%) for generation and display of the disparity map.

We have recently constructed a hybrid pyramid vision machine consisting of the AIS-4000 and 64 transputers [21]. As shown, edge detection and Hough line detection can be executed in less than a second by the pyramid machine. The calculation of costs and the search for dynamic programming can be independently performed for each θ, and, therefore, can be executed in the transputers (one for each θ) to realize a significant speedup.

V. CONCLUSION

This paper presented a method of using dynamic programming in Hough space for solving the correspondence problem in stereo vision for mobile robots in an office environment. The line-based

stereo algorithm described in this paper, can quickly render a non-dense depth map which can be combined with our range-guided stereo system to meet both the far-range and near-range sensing requirements for vision-guided robot navigation. Since most of the objects of interest are man-made, lines are chosen as matching entities for stereo matching. Multiple constraints (uniqueness, epipolar, disparity gradient, figural continuity, and trinocular uniqueness constraints) are integrated in the matching process. The main advantages of this approach are the natural embedding of the constraint of figural continuity, and the speed-up of the matching process. Moreover, it could be used to simultaneously yield certain boundary descriptions in the Hough space which can be very useful for scene analysis.

The time complexity for the search in the dynamic programming in Hough space is $O(pmn)$, where m and n are the numbers of the lines for each θ in the pair of stereo images, respectively, and p is the number of all possible line orientations. Since m and n are usually fairly small numbers, the search is very efficient. In general, line orientations differ (slightly) in the pairs of stereo images which could disturb the order of the peaks in the Hough space. A peak-ordering algorithm is introduced to ensure the optimality of the dynamic programming.

Our preliminary results show that the proposed method works well on several sets of test images of real-world scenes. Relatively small disparity errors are observed and they compare well with other published results. The trinocular method offers even lower error rates and better matches for lines of all orientations. Its execution speed on a Sun SPARC-1 is satisfactory.

The implementation of Hough line detection and dynamic programming has the potential to be parallelized to meet the real-time requirement of robotic applications. One of the future research topics is the extension of this method to matching of 3-D curves of simple parametric forms.

References

[1] D. Marr and T. Poggio, "Cooperative computation of stereo disparity," *Science 194*, pp. 283–287, Oct. 1976.

[2] K. Tate and Z. N. Li, "Depth map construction from range-guided multiresolution stereo matching," *IEEE Trans. Syst. Man Cybern.*, pp. 134–144, 1993.

[3] H. H. Baker and T. O. Binford, "Depth from edge and intensity based stereo," in *Proc. 7th Int. Joint Conf. Art. Intell.*, pp. 631–636, 1981.

[4] Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search using dynamic programming," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, no. 2, pp. 139–154, 1985.

[5] Y. Ohta, T. Yamamoto and K. Ikeda, "Collinear trinocular stereo using two-level dynamic programming," in *Proc. 9th Int. Conf. Pattern Recognition*, pp. 658–662, 1988.

[6] J. E. W. Mayhew and J. P. Frisby, "Psychophysical and computational studies towards a theory of human stereopsis," *Art. Intell.*, vol. 17, pp. 349–385, 1981.

[7] G. Medioni and R. Nevatia, "Matching images using linear features," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 675–685, 1984.

[8] J. H. McIntosh and K. M. Mutch, "Matching straight lines," *Comput. Vision, Graphics, Image Proc.*, vol. 43, pp. 386–408, 1988.

[9] E. Triendl and D. J. Kriegman, "Stereo vision and navigation within buildings," in *Proc. IEEE Int. Conf. Robotics Automation*, pp. 1725–1730, 1987.

[10] P. Kahn, L. Kitchen and E. M. Riseman, "A fast line finder for vision-guided robot navigation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, no. 11, pp. 1098–1102, 1990.

[11] R. O. Duda and P. E. Hart, "Use of the Hough transform to detect lines and curves in pictures," *Commun. ACM*, vol. 15, pp. 11–15, 1972.

[12] P. Burt and B. Julesz, "Modifications of the classical notion of Panum's fusional area," *Perception*, vol. 9, pp. 671–682, 1980.

[13] C. V. Stewart and C. R. Dyer, "The trinocular general support algorithm: A three-camera stereo algorithm for overcoming binocular matching errors," in *Proc. 2nd Int. Conf. Comput. Vision*, pp. 134–138, 1988.

[14] N. Ayache and F. Lustman, "Trinocular stereo vision for robotics," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, no. 1, pp. 73–85, 1991.

[15] U. R. Dhond and J. K. Aggarwal, "A cost-benefit analysis of a third camera for stereo correspondence," *Int. J. Comput. Vision*, vol. 6, no. 1, pp. 39–58, 1991.

[16] W. Hoff and N. Ahuja, "Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 2, pp. 121–136, 1989.

[17] A. Rosenfeld, J. Ornelas and Y. Hung, "Hough transform algorithms for mesh-connected SIMD parallel processors," *Comput. Vision, Graphics, Image Proc.*, vol. 41, pp. 293–305, 1988.

[18] R. Deriche, "Using Canny's criteria to derive a recursively implemented optimal edge detector," *Int. J. Comput. Vision*, vol. 1, pp. 167–187, 1987.

[19] F. Tong and Z. N. Li, "On improving the accuracy of line extraction in Hough space," *Int. J. Pattern Recognition Art. Intell.*, vol. 6, no. 5, pp. 831–847, 1992.

[20] R. Mohan, G. Medioni and R. Nevatia, "Stereo error detection, correction and evaluation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 2, pp. 113–120, 1989.

[21] Z. N. Li and D. Zhang, "Fast line detection in a hybrid pyramid," *Pattern Recognition Letters*, vol. 14, no. 1, pp. 53–63, 1993.

Improving Learning of Genetic Rule-Based Classifier Systems

Alastair D. McAulay and Jae Chan Oh

***Abstract*—A genetic classifier system is reviewed and used for learning rules for classification. Two new strategies are described that enable all the letters of the alphabet to be learned. A "remembering" strategy locks in good rules to overcome forgetting that otherwise occurs during learning. A "specializing" strategy fine tunes the search process for rules. Experiments and an encoding scheme are described. Results show, for the first time, that a genetic classifier-type system can learn to classify all the letters of the alphabet. Further, computer experiments show that the new strategies result in faster and more robust classification involving images of varying position, size, and shape.**

I. Introduction

Learning systems involve searching a given knowledge domain for an optimal feasible solution [14]. In practice, researchers are generally satisfied with learning techniques that provide an adequate suboptimal feasible solution. Many search functions or algorithms converge into local minima which often do not provide an adequate solution. Neural networks, logical deduction, function optimization, simulated annealing, and other mathematical and statistical methods are examples of learning techniques.

Manuscript received August 30, 1991; revised March 5, 1993.

A. D. McAulay is with the Department of Electrical Engineering and Computer Science, Lehigh University, Bethlehem, PA 18015.

J. C. Oh is with the Department of Computer Science and Engineering, Wright State University, Dayton, OH 45435.

IEEE Log Number 9212939.