Efficient Registration of stereo images by matching graph descriptions of edge segments

Nicholas Ayache, Bernard Faverjon

▶ **To cite this version:**

Nicholas Ayache, Bernard Faverjon. Efficient Registration of stereo images by matching graph descriptions of edge segments. International Journal of Computer Vision, 1987, pp.107–131. 10.1007/BF00123161 . inria-00615533

HAL Id: inria-00615533
https://inria.hal.science/inria-00615533

Submitted on 19 Aug 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

International Journal of Computer Vision, 107–131 (1987)
© 1987 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands

Efficient Registration of Stereo Images by Matching Graph Descriptions of Edge Segments

NICHOLAS AYACHE AND BERNARD FAVERJON
Institut National de Recherche en Informatique et Automatique (INRIA) Domaine de Voluceau, BP 105 F-78153 Le Chesnay Cedex, France

Abstract

We present a new approach to the stereo-matching problem. Images are individually described by a *neighborhood graph* of line segments coming from a polygonal approximation of the contours. The matching process is defined as the exploration of the largest components of a *disparity graph* built from the descriptions of the two images, and is performed by an efficient prediction and propagation technique. This approach was tested on a variety of man-made environments, and it appears to be fast and robust enough for mobile robot navigation and three-dimensional part-positioning applications.

1 Introduction

A way to capture the three-dimensional (3D) geometry of a scene is to process a pair of stereoscopic images of this scene. The primary step of any stereoscopic process is to match homologous points between images, that is, points that are images of the same physical object point. If the stereoscopic system was previously calibrated, it is then possible to compute for each pair of homologous image points the 3D coordinates of the corresponding object point in the scene.

To solve the stereo-matching problem, several approaches have been proposed, including (among others), models of human stereovision [1–3], dynamic programming techniques [4–8], relaxation techniques [9–13], correlation techniques [14–20], or structural constraints [21]. For a good review of existing techniques, one can refer to Poggio and Poggio [22].

In this study, we propose to solve the stereo-matching problem by a new method of prediction and propagation of hypotheses applied to a graph-based description of images (earlier versions of this work have been published [23, 24]). First, both images are reduced to line segment *neighborhood graphs*. Segments come from a polygonal approximation of the contours in the images; they are labeled as neighbors when they meet some distance criterion explained later. From the description of the two images, we define a *disparity graph* whose nodes are pairs of potential homologous segments (each node corresponding to a potential 3D segment) and whose edges connect nodes corresponding to neighboring 3D segments in space. The connected components of this graph, which correspond to real or fictive *smooth 3D surface patches*, are explored using an efficient prediction and propagation algorithm. A simple validation procedure is then applied to keep real matches and remove fictive ones.

We now describe in detail the successive steps of this method and present experimental results. We conclude with a discussion and future extensions of this work.

2 Description of Images

Images are first reduced to their intensity boundaries, called edges. Chains of connected edges are then approximated by linear segments. Finally, neighborhood relations between segments are determined. The eventual description of an image is an adjacency graph whose nodes are segments

with local geometric and intensity-based properties and whose edges define neighborhood relations between segments.

2.1 Extraction of Connected Edge Chains

Chains of edges can be extracted by any suitable method. We have successfully tried two different techniques.

The first one is a simplified zero-crossing extraction [25] developed in our laboratory [26] and already described in another application [27]. First, the original image is convolved with two low-pass filters, whose impulse responses realize 7 × 7 and 3 × 3 local averages. Second, the zero crossings of the image difference are computed and connected into chains of edges. Third, each chain is followed by a program that checks the magnitude of the intensity gradient (computed by a Sobel operator) at each point. If the gradient magnitude is below a threshold over more than a given number of connected edge points, these points are removed from the corresponding contour chain. The result of this process is a set of connected, one-pixel-wide edge chains.

The second technique is a slightly improved version of Canny's edge detector [28] developed by Deriche [29]. The image is convolved by an optimal impulse response yielding x and y gradient images. Then local maxima of the gradient magnitude in the gradient direction are extracted and connected to form chains of edge points. During the connection phase, an hysteresis thresholding is applied to remove spurious edge points.

2.2 Construction of Segments

Each chain of edges is approximated by a set of linear segments. Several techniques can be used. The simplest one is the polygonal approximation by successive splits [30]. We are now using a more sophisticated technique, developed by Berthod, [31] that combines a corner detector and a linear least-squares estimate to locate linear segments with higher accuracy. Each segment is stored with the following list of features:

segment [INDEX, x, y, LENGTH, ANG]

where:

INDEX is an integer which identifies the segment,
x and y are the coordinates of the segment midpoint,
LENGTH is the segment length, and
ANG is the orientation of the segment computed within the interval $[-\pi, \pi]$.

These features are local and exclusively geometric measures, except for the orientation feature, which also contains implicitly the sign of the intensity difference (contrast) computed across the segment. It is possible to include additional local properties such as a measure of the average intensity or gradient, or a texture measure computed on each side of the segment.

2.3 Neighborhood Relations and Buckets

In addition to the local features computed for each segment, we define some neighborhood relations between segments. To do so, we first define buckets of segments on the image. These buckets will be used not only to determine the neighborhood of segments at low cost, but also to reduce drastically the computing time devoted to the prediction and propagation of hypotheses (on the use of bucketing techniques, see Knuth [32], for instance).

The computation of the buckets is performed by an algorithm whose complexity is only linear with respect to the number of segments in the image. The overall image is partitioned into m^2 square windows W_i. To each window W_i is attached a bucket b_i that is a list of segments $\{S_j\}$ intersecting W_i. For each segment S_k of the image, the program computes the list of windows $\{W_j\}$ intersected by S_k. First, this list is stored and attached to S_k and, second, the segment S_k is appended to each bucket list b_i attached to the windows W_i (these lists are initially empty). When all segments S_k have been considered, the process ends up with:

1. A list of intersected windows $\{W_i\}$ attached to each segment.
2. A list of intersecting segments $\{S_j\}$ attached to

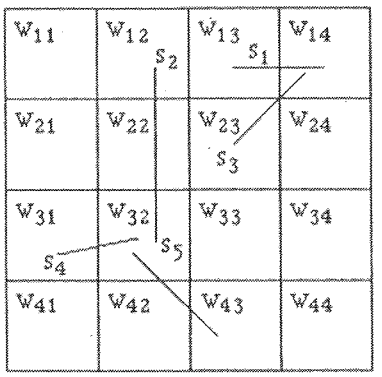


Fig. 1. Computation of buckets and neighborhoods.

each window.

The neighborhood of a segment S_k is defined as the list of segments that intersect at least one common window with S_k. In practice, this list of neighbors is now simply obtained as the union of the buckets S_i attached to the windows W_i attached to S_k. This list of neighboring segments is actually computed for each segment and added to its description record. Figure 1 shows an example of computation of buckets and line segments neighborhoods. In this example, we have the following lists:

List of intersected windows:
S_1: $\{W_{13}, W_{14}\}$; S_2: $\{W_{12}, W_{22}, W_{32}\}$;
S_3: $\{W_{13}, W_{14}, W_{23}, W_{24}\}$; S_4: $\{W_{31}, W_{32}\}$;
and S_5: $\{W_{32}, W_{42}, W_{43}\}$.

List of nonempty buckets:
W_{12}: $\{S_2\}$; W_{13}: $\{S_1, S_3\}$; W_{14}: $\{S_1, S_3\}$;
W_{22}: $\{S_2\}$; W_{23}: $\{S_3\}$; W_{24}: $\{S_3\}$;
W_{31}: $\{S_4\}$; W_{32}: $\{S_2, S_4, S_5\}$;
W_{42}: $\{S_5\}$; and W_{43}: $\{S_5\}$.

List of neighbors:
S_1: $\{S_3\}$; S_2: $\{S_4, S_5\}$; S_3: $\{S_1\}$;
S_4: $\{S_2, S_5\}$; and S_5: $\{S_2, S_4\}$.

Of course, it is possible to use overlapping windows to compute the neighborhoods.

Buckets will also be used to reduce the computing time devoted to the generation and propagation of hypotheses. As we shall see later, these processes are required to match a segment within a given disparity window. By first computing the buckets that do intersect this window, one can substantially reduce the set of potential matching candidates to the union of these buckets. Concerning the window size, it has to be adapted to the disparity window; to generate hypotheses, we use the relatively large (50×50 pixels^2) square-window-based buckets used to compute the line segment neighborhoods. To propagate hypotheses, the disparity window is much narrower, and we have to compute another set of buckets based on vertical rectangular narrow windows. In these last buckets, the segments are sorted accordingly to their orientation, which is done at the expense of a *unique* global sorting of the whole set of segments. This sorting of buckets is also used to speed up the propagation process again, as explained later.

3 Stereo-matching Constraints

In this section, we express the constraints of the stereo-matching problem. When the primitives to be matched are single points, the initial constraints are that homologous points must lie on corresponding epipolar lines and satisfy some local similarity constraints (intensity, gradient). To help solve the correspondence problem, global constraints are usually used: the uniqueness and the continuity constraints [1].

The uniqueness constraint states that a point in one image has at most one correct match in the other image. The continuity constraint expresses the fact that a correct match is generally surrounded by matches having similar disparities (disparity is defined as the difference in position between a pair of matched points).

We now extend those constraints to the case of line segment primitives.

3.1 Local Constraints

We define potential homologous segments as being a pair (L, R) of segments of the left and right images verifying: (1) the epipolar constraint and (2) geometric similarity constraints.

3.1.1 The Epipolar Constraint. Given a point I_1 in one image, its homologous point I_2 is constrained

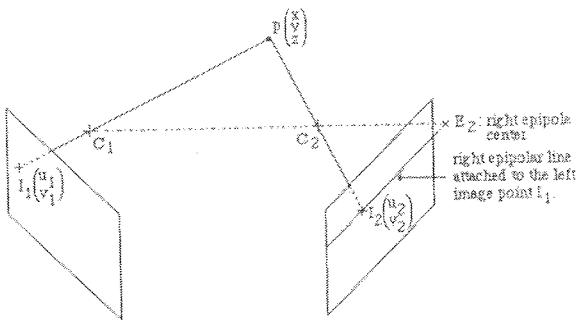


Fig. 2. Geometry of the epipolar lines in the right image: C_1 and C_2 are the optical centers of the cameras.

to lie within a line in the other image called the epipolar line of I_1 (see figure 2). Many authors make the assumption that the optical axes of the cameras are parallel in a plane orthogonal to the image planes, which constrains the search for homologous points along parallel image scan-lines. This constraint can be hard to achieve in practice. Also it can be desirable to make the optical axes to converge to increase the intersection of the fields of view of the cameras and/or improve the accuracy of the stereo reconstruction. Moreover, when the image planes are vertical, because roughly half of the edges in indoor scenes are close to horizontal lines, it is highly desirable *not* to have epipolar lines parallel to the image scan-lines. Therefore we prefer to take into account the actual geometry of the stereoscopic system. It can be done by a very simple and efficient off-line calibration procedure described in appendix A. [33].

To extend the epipolar constraint from points to segments, we require that homologous segments contain at least a pair of homologous points. To simplify, we only require that the midpoint of L have an homologous point on R or that the midpoint of R have an homologous point on L.

This definition is *not* symmetric, and differentiates left-to-right (L, R) and right-to-left (R, L) matches. The advantages of such a definition are twofold:

1. A given segment S has at most one correct match if we assume that the uniqueness constraint is satisfied for points (as there is only one homologous point corresponding to the midpoint of S).
2. Homologous chains of edges that are approximated by a different number of segments in the two images can nevertheless be matched globally by the union of left-to-right and right-to-left matches as is shown in figure 3: here, for instance, we could find the left-to-right matches (L_1, R), (L_2, R), and (L_3, R) and the right-to-left match (R,L_2).

Disparity between potential homologous segments. Although the epipolar constraint is not symmetric, we define a symmetric measure of disparity DISP attached to a pair (L, R) of homologous segments belonging respectively to the left and right images. First, the program computes the common intersection of the segments with epipolar planes. This is done by computing the two epipolar lines in the right image corresponding to the endpoints of the left segment L. These lines intersect the line supporting the right segment R, yielding a segment R′. The intersection R″ of R and R′ is the set of points of R having potential homologous points on L. The corresponding set L″ in L can also be computed. The midpoint I_L of L″ and its homologous point I_R on R″ are computed (I_R is usually not the midpoint of R), and the disparity DISP is defined by (cf. figure 4):

$$\text{DISP} = E_2I_R - E_1I_L$$

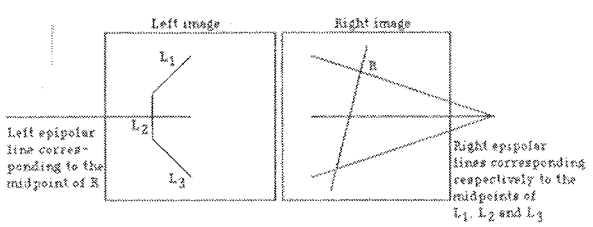


Fig. 3. Left segments L_1, L_2 and L_3 are matched to R while right segment R is matched to L_2.

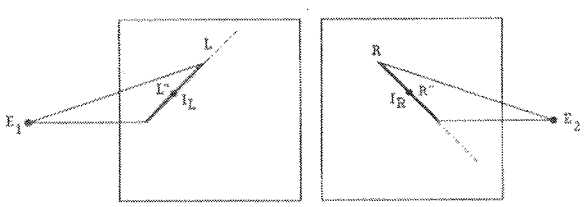


Fig. 4. Definition of disparity between L and R: DISP = $\mathbf{E_2I_R} - \mathbf{E_1I_L}$.

where E_2I_R (resp. E_1I_L) is the distance between I_R and the right epipole center (resp. I_L and the left epipole center).

3.1.2 Geometric Similarity Constraints. The similarity measures consists in computing an error measure between the features LENGTH and ANG attached to L and R, and to reject the potential match (L, R) whenever one of these error measures is above a threshold.

These thresholds are set relatively tight during the prediction steps and are loosened during the propagation step. They are computed in the following manner:

—Length is a relatively unstable feature. This is mainly due to the poor robustness of the polygonal approximation scheme. Typically a length ratio between homologous segments of 1.5 in the prediction step or 3 in the propagation step is tolerated.

—Angle is a robust feature with respect to the polygonal approximation scheme. Therefore the angular disparity between two homologous segments primarily comes from the viewpoint difference. A statistical study performed by Arnold and Binford [34] showed that, with narrow angle stereo—i.e., when the viewpoints are relatively close—the statistical distribution of the angular disparity between homologous segments (images of randomly oriented 3D segments) is sharply centered close to zero. Therefore we set a tight threshold for the angular disparity (typically 15 degress during the prediction and propagation steps).

3.2 Global Constraints

3.2.1 The Continuity Constraint. Up to now, we have only applied local geometric constraints to define potential homologous segments. We introduce now a global constraint known as the continuity constraint (cf. Marr and Poggio [1]): The observed scene is made of physical objects whose surfaces vary smoothly in general. Therefore, disparity between homologous points in the images should also vary smoothly in general, except at a few depth discontinuities. Hence, if a match (L, R, DISP) is correct, it is very likely that most of the neighbors L′ of L (resp. R′ or R) can find a match R′ (resp. L′) with a disparity close to DISP, which should not be the case when the initial match is incorrect. In other words, applying the continuity constraint to a given match will yield a large number of likely correct matches within the neighborhoods of L and R if the intial match is correct, and a small number of likely incorrect matches otherwise. If this process is applied repeatedly to all the new matches obtained, it should end up with a much larger final number of matches when the initial hypothesis is correct than when it is incorrect. This fact can be expressed in terms of large or small connected components of the disparity graph defined below.

Definition of the disparity graph. The disparity graph is defined as follows: (1) nodes are pairs (L, R) of potential matches between left and right images; and (2) edges connect pairs of nodes (L, R), (L′, R′) such that L′ and R′ are respective neighbors of L and R in the image descriptions, and such that the disparity gradient between (L, R) and (L′, R′) (computed as the difference between the disparities attached to each match) is lower than a locally computed limit, called the *disparity gradient limit*.

The disparity gradient limit is computed as a *function of the position of the reconstructed corresponding 3D point*, and it is adapted to correspond to a *constant tolerated variation of depth* ϵ in space between two neighboring reconstructed segments (this definition differs from the one of Mayhew and Frisby [12] and Pollard et al. [13]). Computation details are in appendix B.

A connected component of the disparity graph corresponds to a subset of 3D points that belong to what we call a *smooth surface patch*.

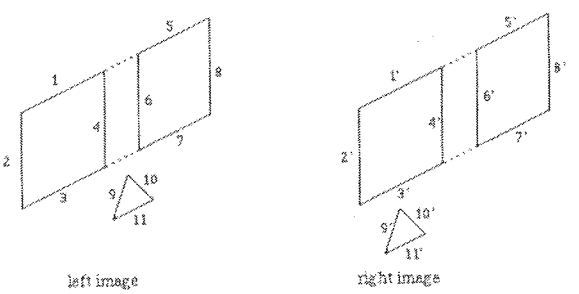


Fig. 5. The left and right images of a triangle lying in front of two rectangles.

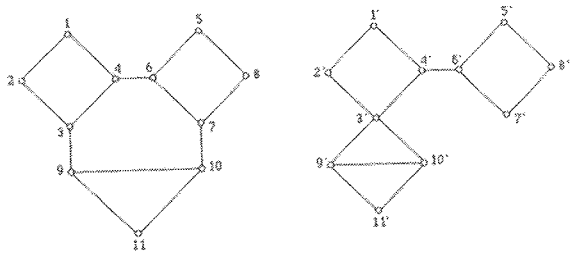


Fig. 6. Neighborhood graphs

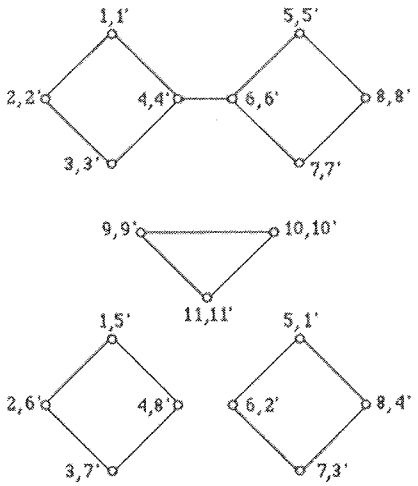


Fig. 7. Disparity graph.

As an illustration of this definition, let us consider the example shown in figure 5. There is a triangle lying below and in front of two rectangles. Figure 6 shows the neighboring graph produced in the two images. Figure 7 shows the disparity graph obtained for a given value of ε. There are four connected components.

The first one corresponds to the matching of the two rectangles against their respective homologues. The second one corresponds to the matching of the triangle. There is no connection between (3, 3′) and (9, 9′) or between (7, 7′) and (10, 10′) because the discontinuity in depth between the triangle and the rectangles is such that the corresponding 3D points are at a distance greater than ε.

The third and fourth components of the disparity graph correspond to "phantom" images of the rectangles obtained by matching each of them with its *opposite* homologue.

3.2.2 The Uniqueness Constraint. To enforce the uniqueness constraint, we must keep only one match per left or right segment. In case of conflict, the correct matches will usually correspond to the larger connected component. This gives us a criterion to choose between conflicting matches.

In the following sections, we describe the practical algorithm used to achieve the stereo matching of line segments.

4 Prediction and Propagation Algorithm

To compute the connected components of the disparity graph, we use a prediction and propagation algorithm. The idea is to generate some potential pairs of homologous segments based on a strong local similarity of contours, and then to explore the attached connected components by a recursive propagation.

4.1 Prediction of Hypotheses

Initially, the potential disparity range within the image is relatively large, and depends on the geometry of the imaging system and of the observed scene. (Typically the disparity range is about one-third of the image width.) When this interval has been estimated, the role of the prediction program is to establish a list of tentative matches (R, L) with associated disparities DISP lying within this interval and verifying the local constraints defined in the previous section.

In order to reduce the number of false matches, only segments that are neither too small (poor estimation of the orientation) nor too long (likely to be broken in the other image) and with orientations not too close to the orientation of the epipolar lines are considered.

The hypotheses are generated with some degree of uniformity over the whole image by randomly selecting the right segment R, and by stopping the prediction process when the number of matches is large enough. Also, the search for a left candidate L is substantially accelerated by the use of the squared buckets previously defined: given a right segment R, the program computes in the left image, the epipolar line corresponding to the midpoint of R, and on this line,

the segment S_{disp} associated to the interval of potential disparities. The matching candidates L are then selected among the union of the left buckets intersected by the segment S_{disp}.

4.2 Propagation Algorithm

At the end of the prediction stage, there are a certain number of potential matches. Each match corresponds to a single node in the disparity graph. The idea is to use the disparity graph to propagate these matches within their neighborhood to recover subsets of 3D segments lying on a same smooth surface patch (that is, matches that belong to the same connected subgraph of the disparity graph).

A conflict is defined as a pair of matches (L, R), (L, R′) such that R ≠ R′ or (L, R), (L′, R) such that L ≠ L′. Conflicts occurring during the propagation stage are immediately solved to avoid multiple matches within the same connected subgraph. Conflicts occurring between two distinct connected subgraphs are solved after the propagation of each connected component by analyzing their respective size (power of prediction).

4.2.1 Exploration of Connected Components. The propagation alogrithm starts from a prediction, that is, a given node of the disparity graph. All the nodes connected to this node in the disparity graph are then recursively considered. In practice, the disparity graph is constructed during this exploration by alternatively and recursively looking for a match for the neighbors of the left or right segments in the left or right images. The propagation process is achieved by the two following very simple recursive procedures, called PROPAGATE_LEFT and PROPAGATE_RIGHT:

Recursive procedure PROPAGATE_LEFT (segment_left, predicted_disparity)
- if ALREADY_VISITED (segment_left) then return;
- (segment_right, actual_disparity): = MATCH_LEFT (segment_left, predicted_disparity);
- if (segment_right = nil) then return.
- For each neighbor neighbor_right of segment_right do
  PROPAGATE_RIGHT (neighbor_right, actual_disparity); end for;

end procedure PROPAGATE_LEFT.

Recursive procedure PROPAGATE_RIGHT (segment_right, predicted_disparity)
- if ALREADY_VISITED (segment_right) then return;
- (segment_left, actual_disparity): = MATCH_RIGHT (segment_right, predicted_disparity);
- if (segment_right = nil) then return.
- For each neighbor neighbor_left of segment_left do
  PROPAGATE_LEFT (neighbor_left, actual_disparity); end for;

end procedure PROPAGATE_RIGHT.

To propagate a right-to-left hypothesis (R, L, DISP), the program calls the sequence PROPAGATE_LEFT (L, DISP); this procedure first checks whether this segment has been visited already. If yes, it stops, in order to prevent an infinite loop. Otherwise the procedure MATCH_LEFT is called. MATCH_LEFT is a procedure very similar to the one used to generate hypotheses. It takes as input a left segment L and a predicted disparity PREDICTED_DISPARITY, and returns a match (L, R′) with actual disparity ACTUAL_DISPARITY. To do so, this program selects, among the right segments, those whose disparity with L is within the interval PREDICTED_DISPARITY + Δ1, PREDICTED_DISPARITY + Δ2 where Δ1 and Δ2 are computed as a function of the position in space of the reconstructed 3D segment (see appendix B).

Here again, the use of precomputed narrow vertical buckets drastically reduces the number of candidates as potential matches in the right image. Also, the initial sorting of the buckets with respect to the segment orientations allows for a fast binary search within each bucket of the candidates whose orientations are compatible with the orientation of the left segment.

If several candidates remain, only the one with the disparity closest to PREDICTED_DISPARITY is retained. By doing this, we privilege segments belonging to the surface whose distance from the observer has the smallest variation (frontoparallel surfaces). Alternative criteria could be used at this point, for instance, choosing the match yielding the 3D segment closest in the scene to the pre-

vious one. Anyhow, the current criterion, which yields good results in our experiments, has the merit of being simple and computationally cheap.

If no match is found, the program MATCH_LEFT returns NIL. In this case, the procedure PROPAGATE_LEFT stops, which means that it cannot propagate matches any further with the predicted disparity PREDICTED_DISPARITY and the tolerances $\Delta 1$ and $\Delta 2$ applied to left segment L.

When MATCH_LEFT returns a match R′ and an actual disparity ACTUAL_DISPARITY, the procedure PROPAGATE_LEFT calls the symmetric procedure PROPAGATE_RIGHT for each neighbor of R′ and with an updated predicted disparity ACTUAL_DISPARITY.

The result of this propagation process is, for each hypothesis, a set of matches (L, R, DISP) corresponding to what we called a smooth surface patch. In the example of figure 5, the hypothesis (1, 5′) would propagate within the third connected component of the disparity graph of figure 7, yielding the matches (2, 6′), (3, 7′), (4, 8′), while the hypothesis (1, 1′) would propagate within the first component, yielding the matches (2, 2′), (3, 3′), (4, 4′), (5, 5′), (6, 6′), (7, 7′), (8, 8′).

4.2.2 Pruning of Hypotheses. Because the technique used to select a match between conflicting matches during the propagation process is a *best first choice* technique, the result of the propagation process might depend on the starting node chosen within each connected component.

Nevertheless, in practical experiments, it appeared that this dependency was minor, due to the computation of the disparity gradient limit as a function of the distance. As a consequence, in order to substantially reduce the computing time of the algorithm, it is desirable to propagate only one hypothesis by connected component of the disparity graph. This is achieved by removing after the propagation of an hypothesis, all the forthcoming hypotheses corresponding to a match already propagated.

In the example of figure 5, if the hypotheses (2, 2′), (4, 4′), (6, 6′), (8, 8′) are made by the prediction algorithm, then, after the propagation of any one of these hypotheses, the others will belong to the propagated matches and therefore be discarded.

4.2.3 Conflicts Between Connected Components. When a new hypothesis is propagated, the same algorithm applies, without considering the matches obtained by previous hypotheses. When the propagation is over, conflicts between distinct connected components are solved by discarding the conflicting matches attached to the concerned hypothesis having the smaller *power of prediction* (number of matches). In the example of figure 5, if a given hypothesis yields the matches {(1, 1′), (2, 2′), (3, 3′), (4, 4′), (5, 5′), (6, 6′), (7, 7′), (8, 8′)}, and if a new hypothesis yields the matches {(5, 1′), (6, 2′), (7, 3′), (8, 4′)}, then all the conflicting matches of the second hypothesis (in this case the four of them) will be discarded because they belong to a smaller connected component of the disparity graph (of size 4 instead of 8). As one can see, this is an efficient procedure to distinguish between real objects and "phantoms" in case of repetitive structures.

Due to the potential deletions occurring after each new propagation, the size of an hypothesis can change after the propagation of another hypothesis; nevertheless, the algorithm uses only the initial size of each hypothesis to compare conflicting matches. By doing this, the final result does not depend on the order in which the hypotheses are compared.

5 Final Validation

When all the hypotheses have been propagated, some of the connected components have been totally or partially erased by better ones. However, some false matches not conflicting with others (in the sense of the uniqueness constraint) may remain. In our experiments, such matches represent 5%–10% of the final matches. They usually correspond to isolated 3D segments. Thus, they can be easily removed by keeping only matches that are connected to a minimum number of nodes in the disparity graph. This minimum number is set depending on the nature of the observed scenes. In our experiments, we rejected connected components of size smaller than four. In this case, the remaining false matches represent less than 2% of the matched segments.

6 Experimental Results

The stereo-matching program has been tested on several scenes including outdoor and indoor scenes as well as scenes with industrial parts. We present here the results of the matching process applied to four typical scenes, which are called (1) room, (2) office, (3) corridor, and (4) industrial part, and which are shown in figures 8, 14, 20, and 26, respectively.

6.1 Preprocessing

Pairs of 512 × 512 images are taken by a pair of standard cameras. Contours on each image are then extracted and approximated by linear segments. For the first three scenes, the edge extraction is performed by the gradient maxima technique while, in the fourth scene, the zero-crossing technique is used (see section 2.1). All this preprocessing is independent of the stereo-matching program and could be done in a fraction of a second on specialized hardware. In its current implementation, the preprocessing requires an average of 120 s of CPU time per image on a Sun 3 work station for the gradient maxima technique and 20 s for the zero-crossing technique.

6.2 Stereo Matching

We first present the results concerning the indoor scenes shown in figures 8, 14, and 20. In these indoor scenes, the epipolar lines are tilted and make an angle with the horizontal plane of about 45 degrees. This is obtained by placing the second camera on the right of and below the first one. This is very interesting because it allows for the accurate reconstruction of both vertical and horizontal segments, which represent the majority of the image segments. For each scene, we show respectively in figures 9, 15, and 21 the initial edge segments. Only the segments of length greater than 12 pixels are selected by the stereo-matching program: these segments are shown in figures 10, 16, and 22.

In figures 11, 17, and 23, we show the pairs of segments matched by the stereo-matching algorithm. Homologous segments are labeled with the same number.

In figures 12, 18, and 24, we show, superimposed on the right image segments, the horizontal distance (computed from the optical center of the camera and projected on the horizontal plane) and, in figures 13, 19, and 25, the elevation (computed from the floor) of the midpoints of the reconstructed 3D segments. These measures agree very well with the direct measures we can make on the environment.

A careful examination of the matching results showed the following errors: segment 127 in the room and segments 78 and 22 in the corridor. These mistakes come from the absence of an edge in one of the images, yielding a false match that still agrees with its neighborhood (a node in the disparity graph connected to a sufficient number of other nodes). Of course, one could use the results obtained from several viewpoints to correct some of these errors. We are currently working on this improvement. Also, some of the matches correspond to specularities, which yield virtual 3D segments that can appear above the ceiling, below the ground, or beyond a wall. For instance, segments 17, 18, 19, 21, and 32 of the corridor are virtual images of ceiling lights that appear at the height of the ceiling on the right of the right wall. Additional semantic knowledge could be incorporated in the system to remove these segments. Notice, however, that segments 53 and 54 in the corridor example are measured at the correct negative elevation, due to a slope one can notice in the image. Notice also the fact that, in the six scenes, horizontal segments on windows, on the

Fig. 8. Stereo view of a room.

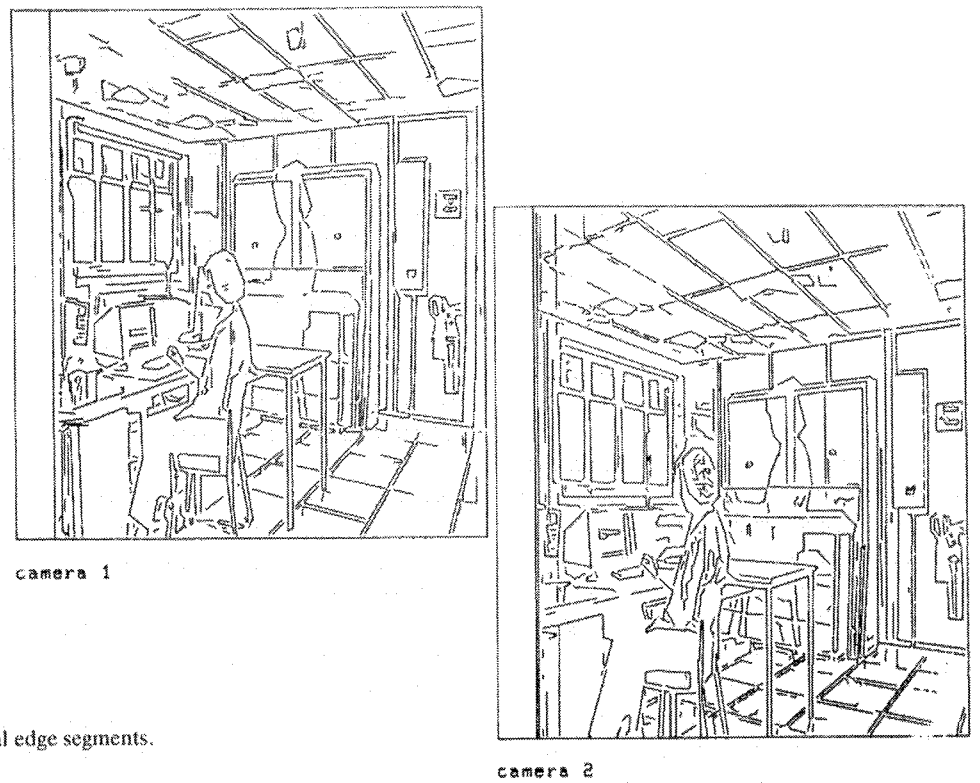


Fig. 9. Initial edge segments.

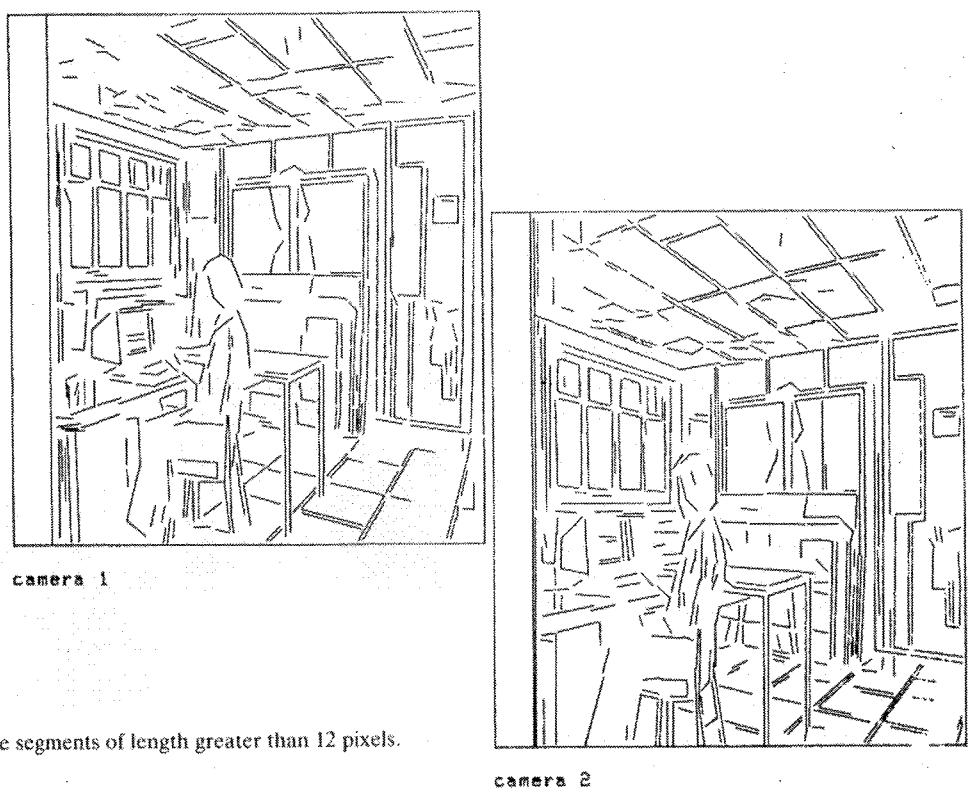
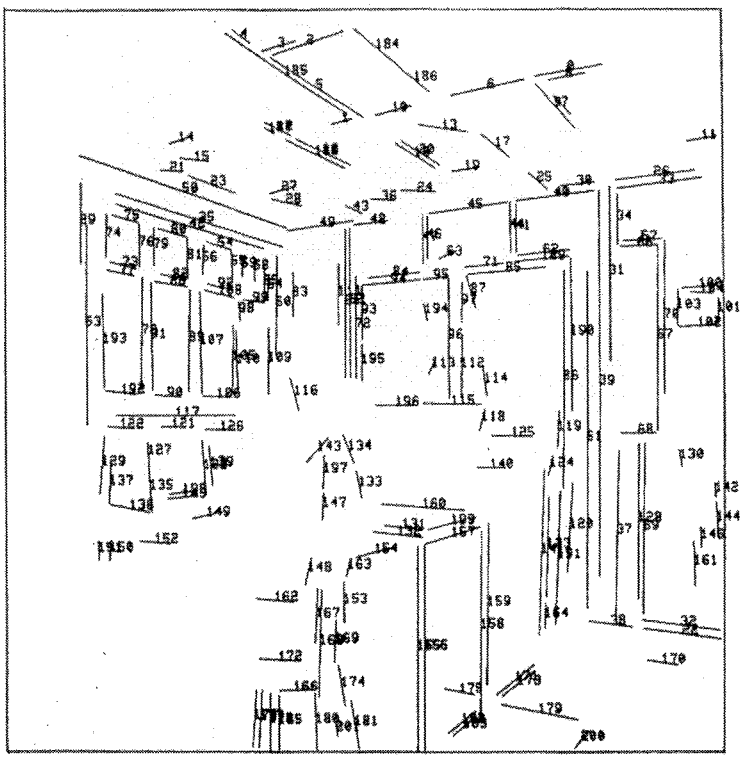


Fig. 10. Edge segments of length greater than 12 pixels.



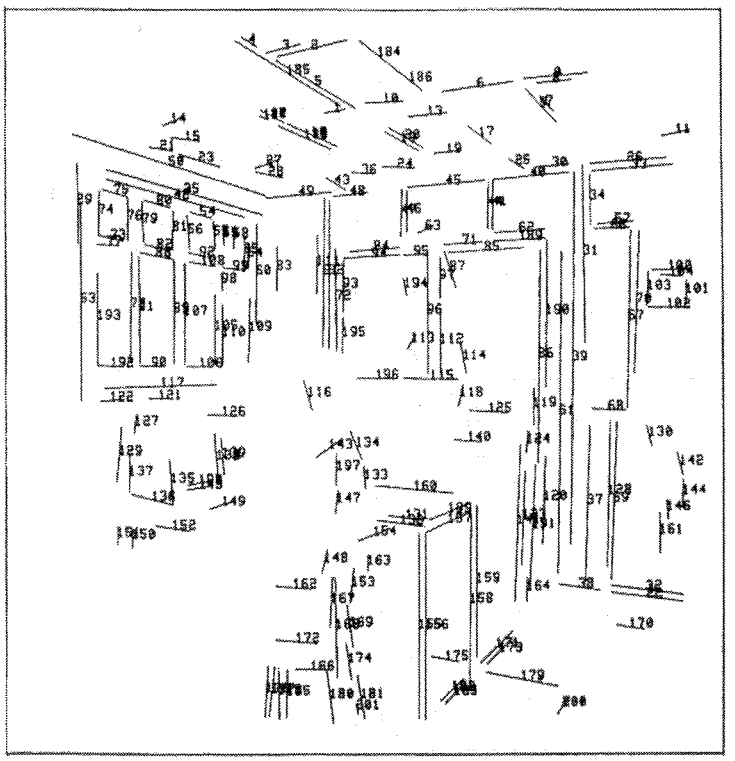


Fig. 11. Matched segments between camera 1(a) and camera 2(b).

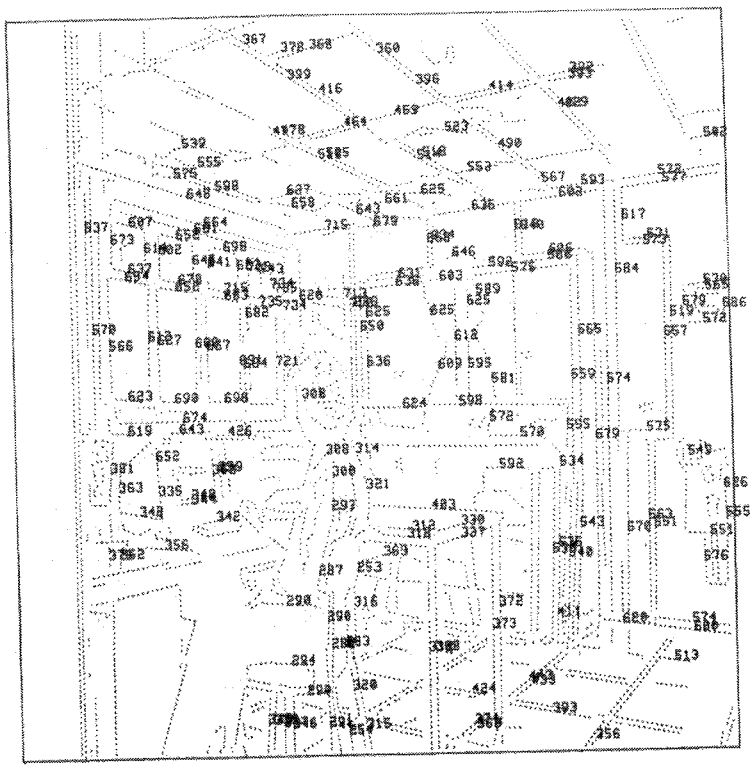


Fig. 12. Horizontal distance (cm).

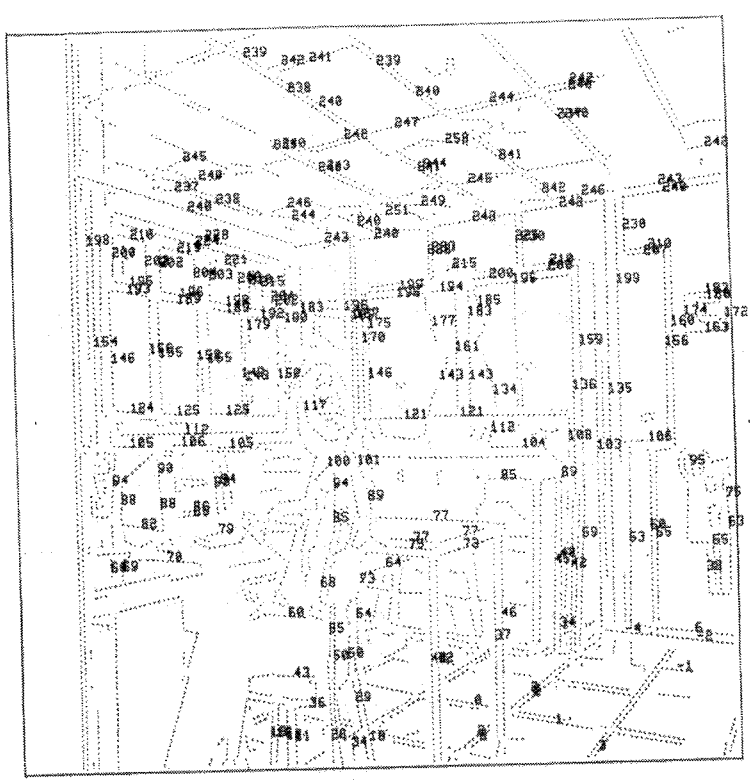


Fig. 13. Elevation (cm).

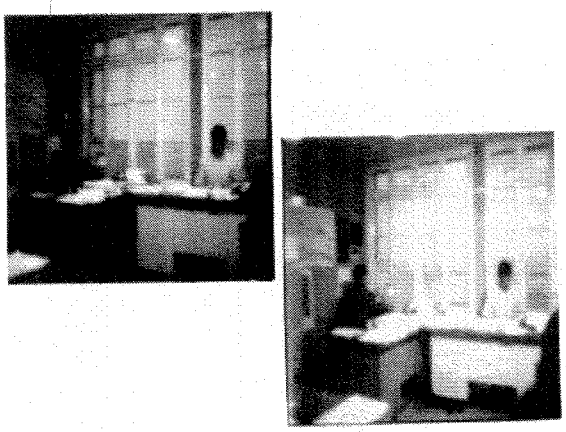


Fig. 14. Stereo view of an office room.

Fig. 15. Initial edge segments.

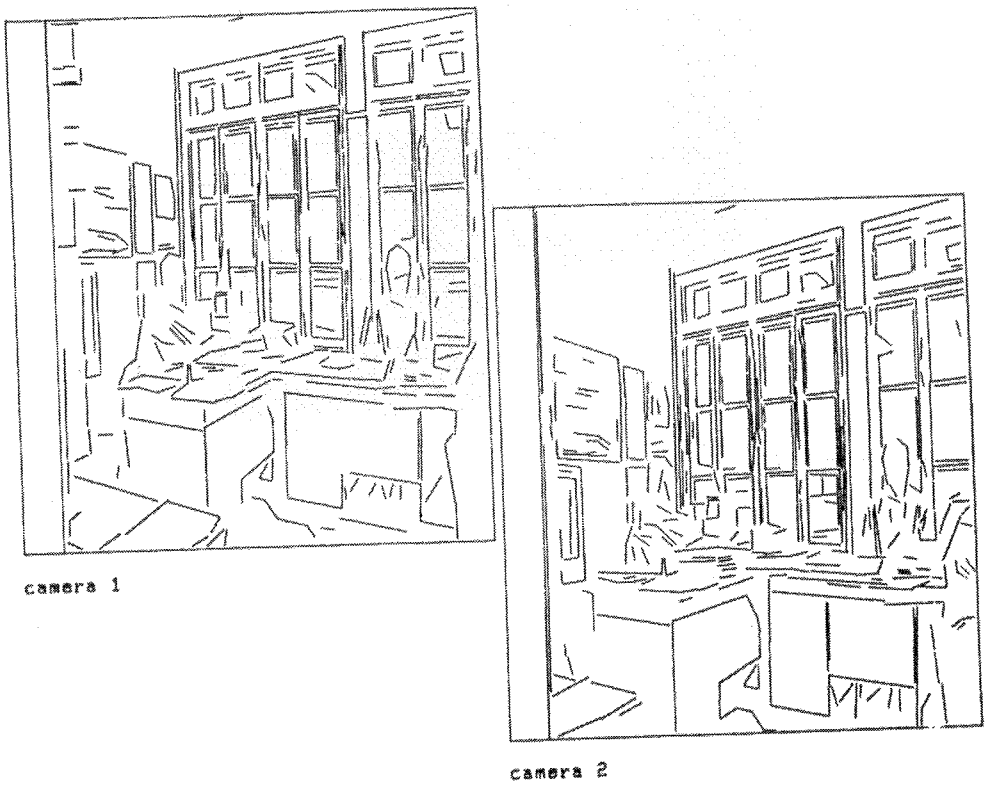


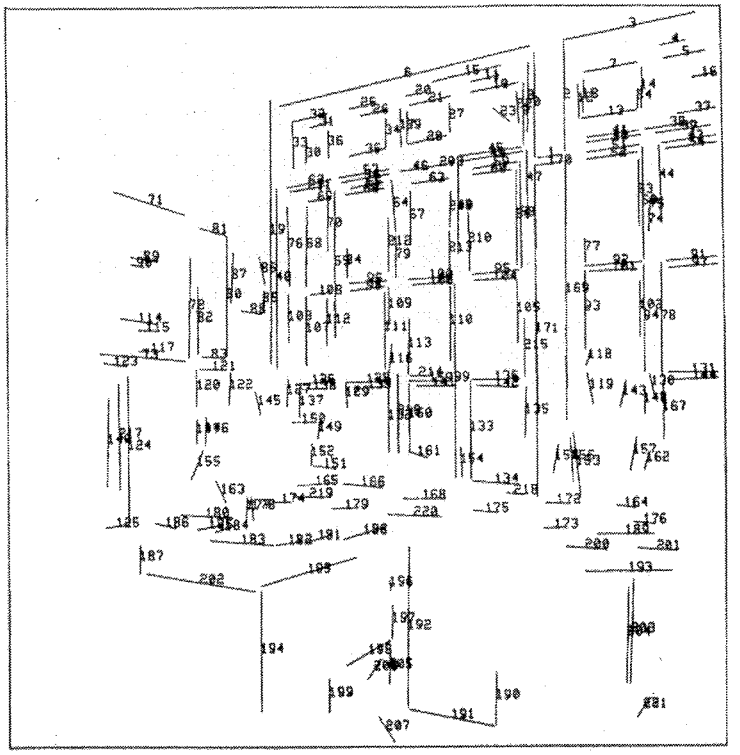
Fig. 16. Edge segments of length greater than 12 pixels.

ground, and on ceiling usually form clusters of segments having the same elevation, and that the computed horizontal distance is sound with the observed scene and allows for an easy detection of the closest obstacles.

The position of the reconstructed 3D segments was used by a geometric matcher developed in our laboratory [35] to compute the actual motion of the vehicle for translations of about half a meter and rotations of about 10 degrees with a good accuracy. These results are detailed in two other reports [36, 37].

In figures 26–29, we show the results obtained on an industrial part. The geometry of the cameras was totally different (larger angle stereo, almost horizontal epipolar lines, teleobjectives instead of large angle lenses). After an automatic calibration of this new system, the same stereo-matching program was used. The results for another position of the object are shown in figures 30–33. In the vertical view of the reconstructed segments (figures 29 and 33), it is interesting to notice the linear clusters corresponding to points that are in a vertical plane, and to observe a horizontal rotation of about 40 degrees of this plane. This qualitative observation was confirmed by a quantitative experiment. The 3D segments were processed by the above-mentioned geometric matcher [35] and most of the midpoint segments present in the two scenes were matched between these two scenes by a solid displacement corresponding to the correct one. The average error between 3D-matched points is 4.5 mm, while the diameter of the object is about 25 cm, which gives a good idea of the quality of the reconstruction.

To conclude this experimental report, we present in table 1 the computing time devoted to the stereo-matching process. Note that the number of matched points is about 20 times the number of matched segments (as the average length of a segment is about 20 pixels). The program is written in a highly recursive style in C and runs on a Sun 3 workstation.



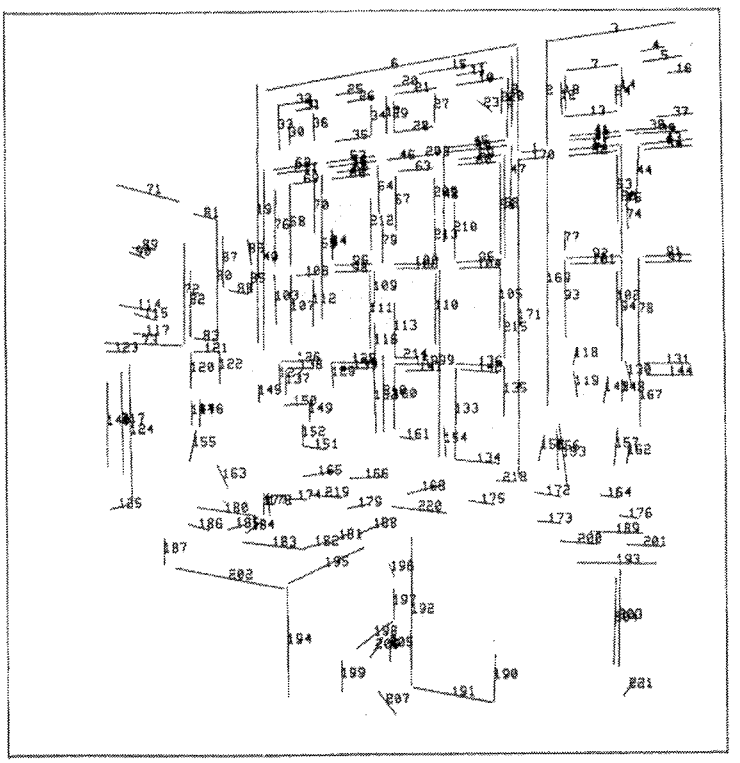


Fig. 17. Matched segments between camera 1(a) and camera 2(b).

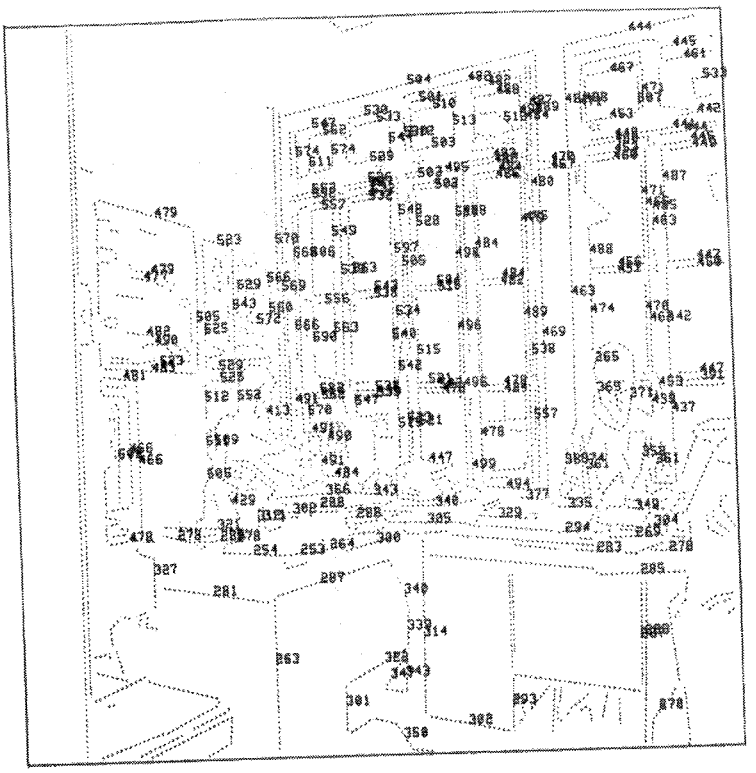


Fig. 18. Horizontal distance (cm).

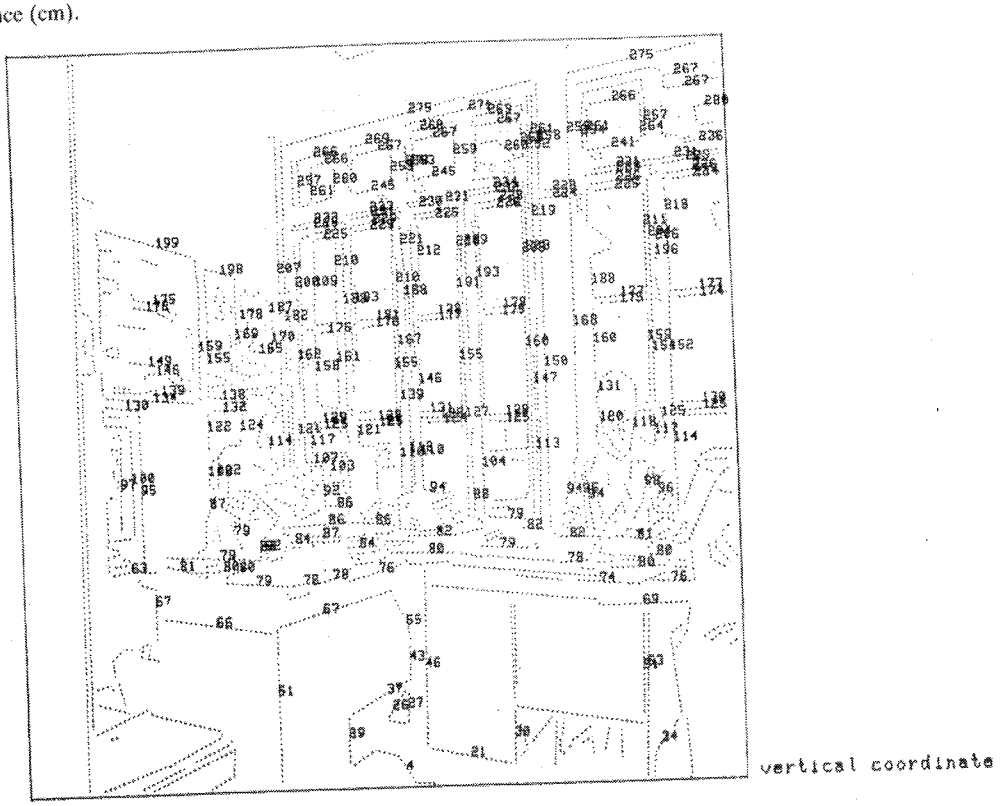


Fig. 19. Elevation (cm).

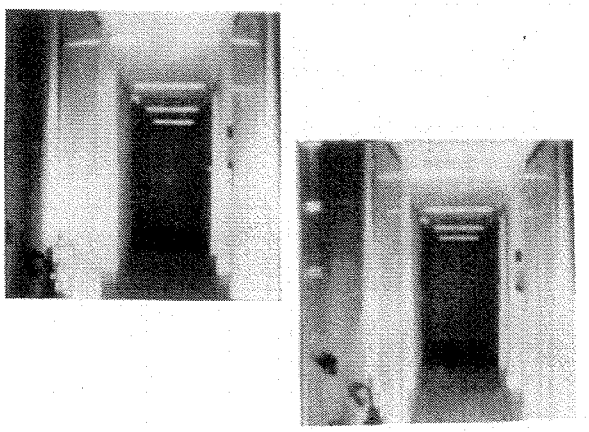


Fig. 20. Stereo view of a corridor.

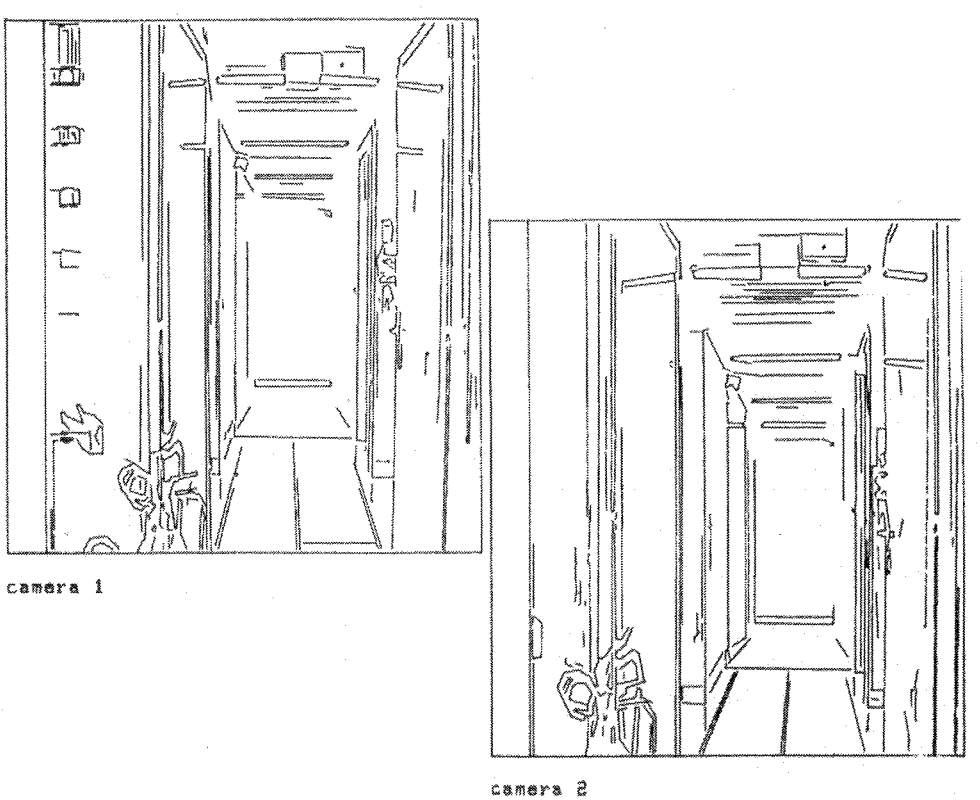


Fig. 21. Initial edge segments.

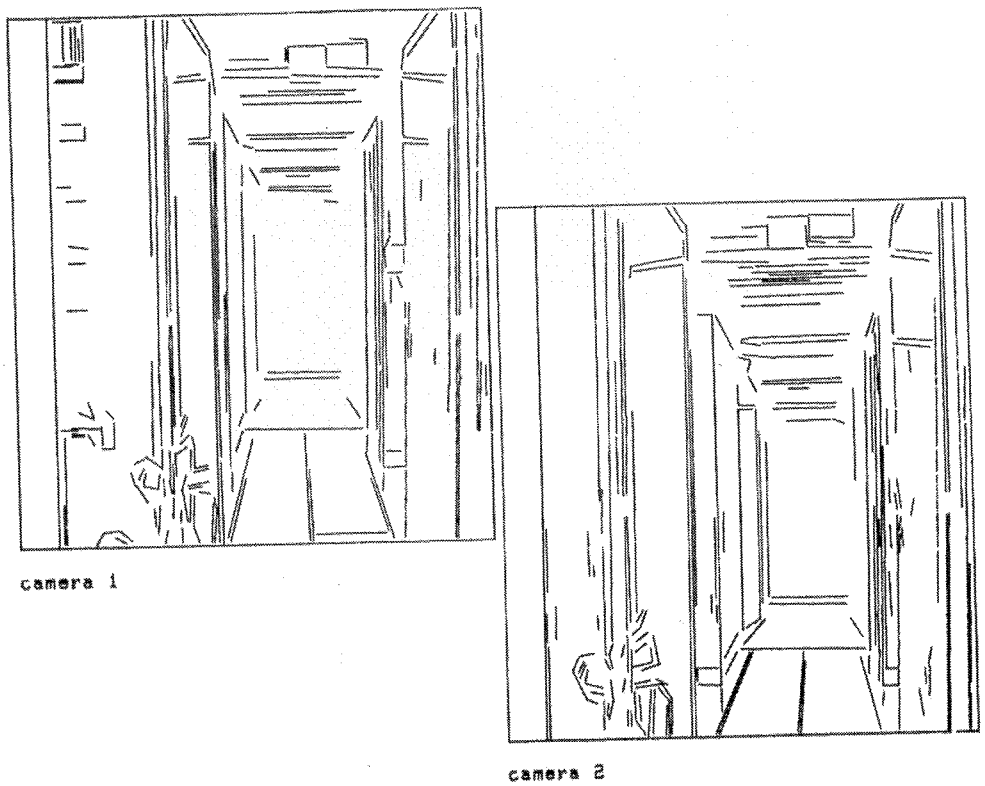


Fig. 22. Edge segments of length greater than 12 pixels.

	Room	Office	Corridor	Industrial part
Number of left segments	385	362	177	334
Number of right segments	401	370	188	374
Number of generated hypotheses	75	75	75	32
Number of pairs of matched segments	202	223	86	150
CPU time, construction of the adjacency graph	1.5 s	1.5 s	0.5 s	1.0 s
CPU time, generation of hypotheses	0.2 s	0.2 s	0.1 s	0.2 s
CPU time, propagation and validation of hypotheses	5 s	5 s	2 s	3 s

Table 1. Computing time of the stereo matching process.

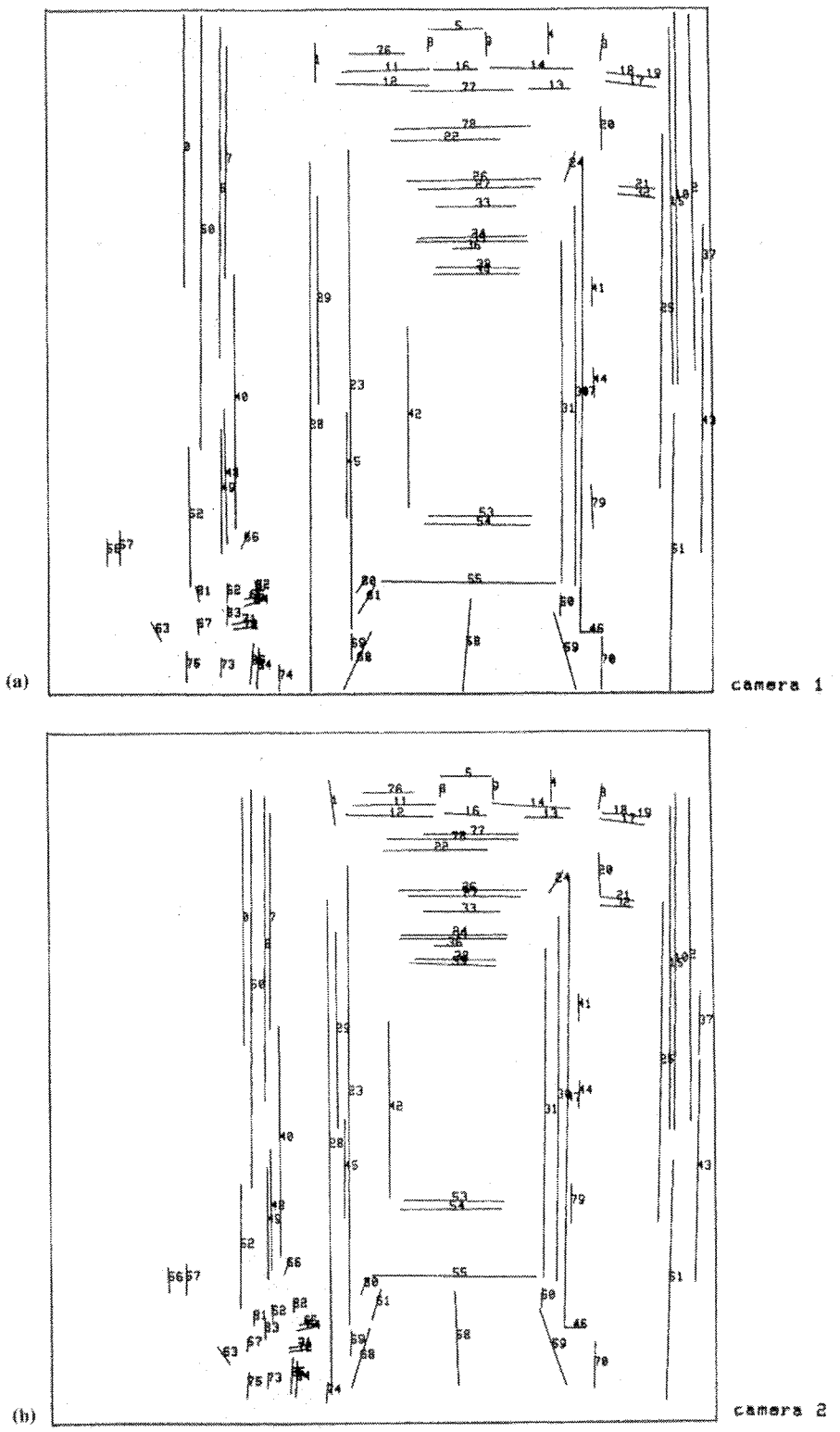


Fig. 23. Matched segments between camera 1(a) and camera 2(b).

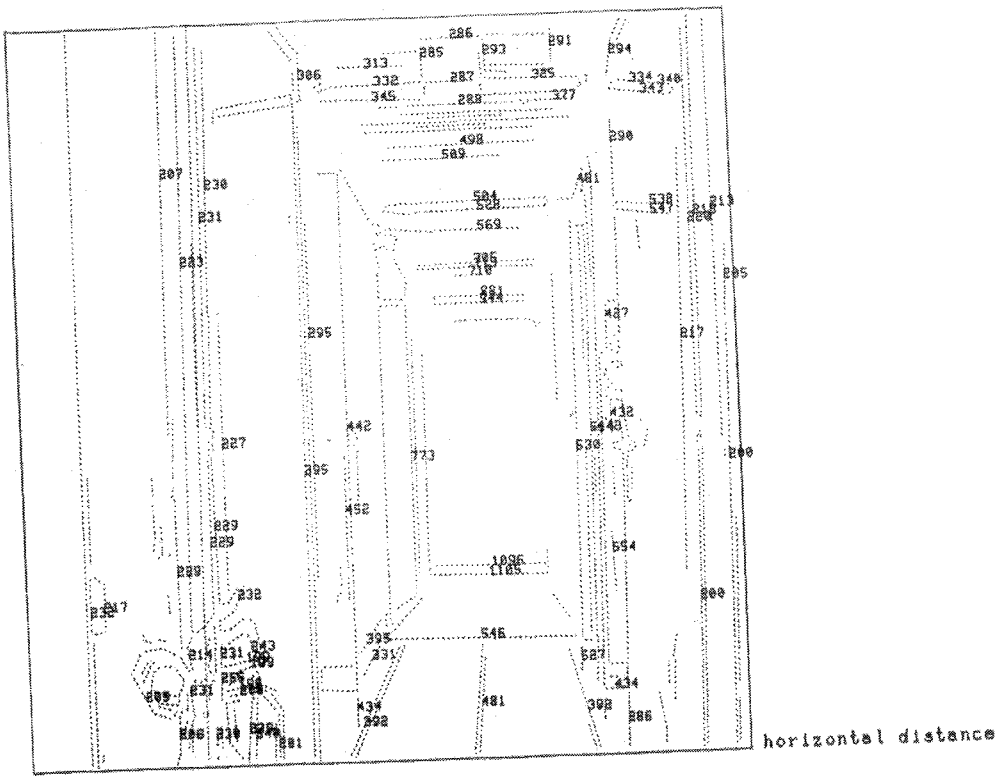


Fig. 24. Horizontal distance (cm).

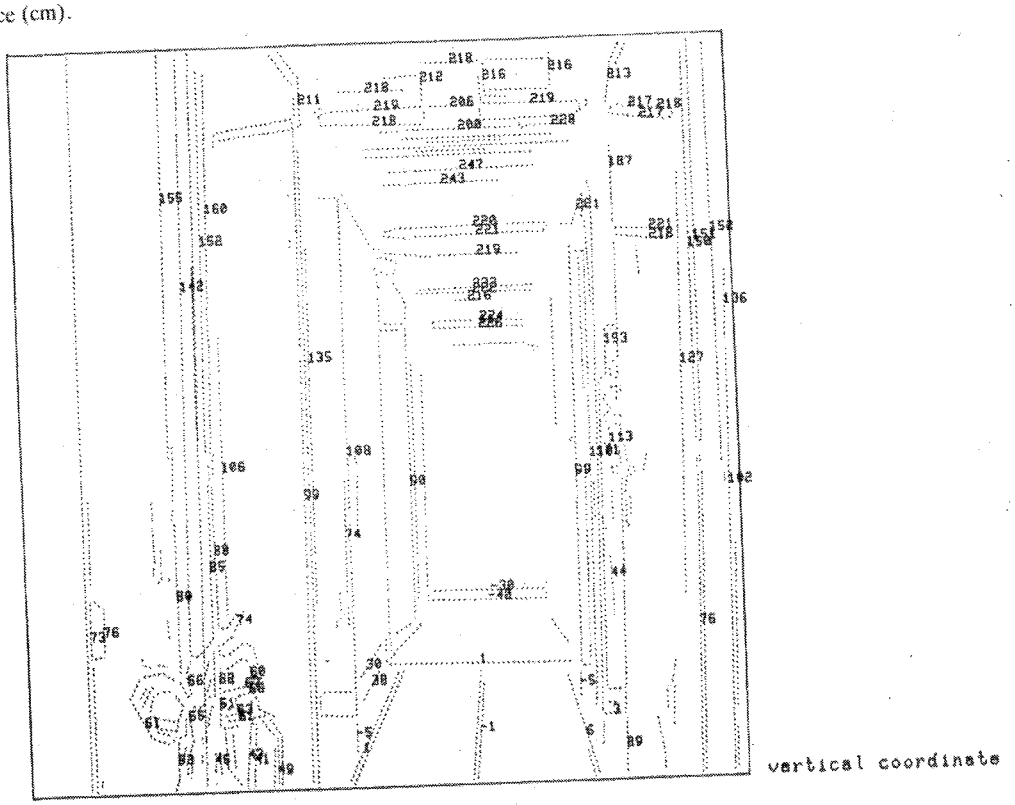


Fig. 25. Elevation (cm).

Fig. 26. Stereo view of an industrial part (position 1).

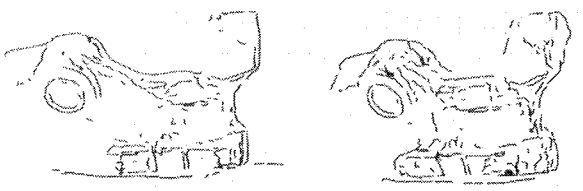


Fig. 27. Initial edge segments.

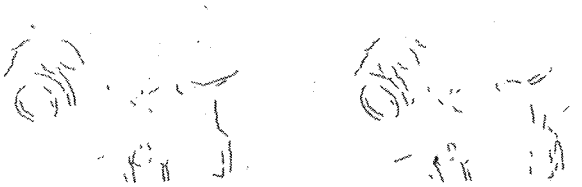


Fig. 28. Matched segments.

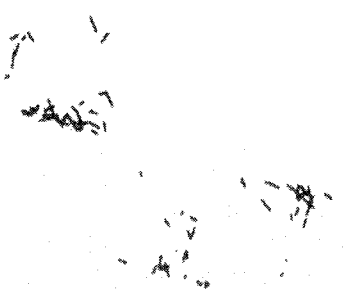


Fig. 29. Vertical view of the reconstructed 3D segments.

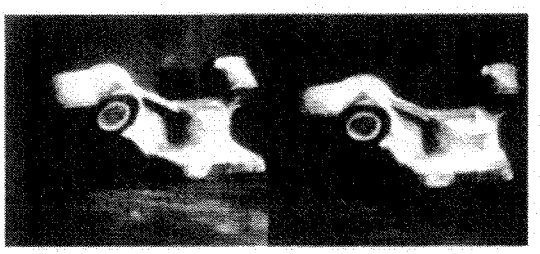


Fig. 30. Stereo view of an industrial part (position 2).

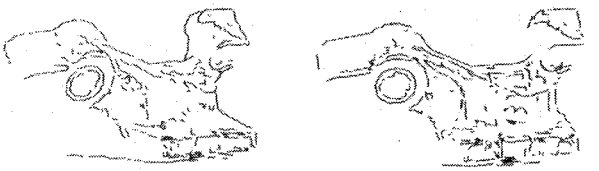


Fig. 31. Initial edge segments.

Fig. 32. Matched segments.

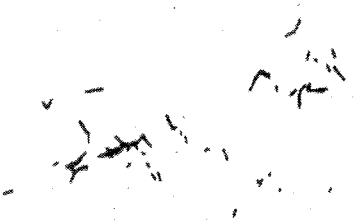


Fig. 33. Vertical view of the reconstructed 3D segments.

7 Conclusion and Discussion

We have presented a new approach to solve the stereo-matching problem using a graph-based description of images and a technique of prediction and propagation of hypotheses within a disparity graph. The method appears to be particularly fast and robust.

The use of a *neighborhood graph* of line segments provides a drastic size reduction of the image representations. As it was shown in the first examples, it is very well adapted to indoor scenes where there are a lot of straight lines, but also to other scenes where most of the contours are curved, as in the industrial-part examples. Also, the fitting of straight lines is done with subpixel accuracy, providing more accurate reconstruction.

The use of the *disparity graph* defining the connectivity between two nodes (L, R) and (L′, R′) each time the distance between the reconstructed 3D segments is smaller than a preset constant ε yields smooth surface patches that are *intrinsic en-*

tities of the 3D scene, remaining unchanged if the point of view or the geometry of the steroscopic system is modified. This is of crucial importance in navigation and recognition applications, where it is necessary to compare visual descriptions obtained from different viewpoints. From a practical point of view, the precomputation of a disparity gradient as a function of the position of points in 3D space enables very efficient computations.

In our implementation, we have almost exclusively used the geometric properties of the line segments in the images. It is interesting to note that they are sufficient for the registration of stereo images. Of course, it is possible to incorporate some additional information to the description of each segment such as intensity-based local features (e.g., average intensity on each side of a segment, average contrast across a segment). The comparison of these additional features could reduce the number of hypotheses and also the number of errors, given that they are stable with respect to the variation of the point of view.

The stereo matcher presented in this report has already been used to demonstrate fast recognition and positioning of 3D objects as well as to complete some navigation tasks for an autonomous vehicle developed in our laboratory. The results are presented elsewhere [35–37].

To conclude, we think that the methodology developed in this application—that is, the use of a symbolic geometric description of the images and of a graph exploration based on a prediction and propagation of hypotheses strategy—is quite general and could probably be applied, by relaxing the epipolar constraint, to achieve fast registration between a time sequence of monocular moving images. Also, the prediction and propagation scheme is very well suited to include this kind of registration process within a closed loop, results obtained at previous stages being used to guide the prediction at the current stage. We are currently working on these problems.

8 Appendix A: Calibration of the Stereoscopic System

8.1 Computation of the Perspective Transformations

Each camera can be modeled by a perspective transformation **T**. This transformation is linear in homogeneous coordinates and can be represented by a 3×4 matrix that maps scene points to image points:

$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = \mathbf{T} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{1}$$

The perspective transformations $\mathbf{T}_1$ and $\mathbf{T}_2$ of the cameras are automatically estimated by the following procedure. A planar grid of precisely located orthogonal straight lines is placed in front of the two cameras in (at least two) different known positions. Images of these lines are extracted, and their equations are accurately computed by a least-squares estimator. Then, the intersection points of the grid are computed. As soon as six different non-coplanar grid points have been registered between the scene and an image, the corresponding perspective transformation **T** is estimated by a least-squares estimate.

In effect, each transformation **T** constrains 11 unknowns (and not 12, as **T** is defined up to a scale factor), and each registration yields two linear equations by elmination of s in system 1. In practice, more than 50 registrations are computed to obtain an accurate estimate of **T**.

8.2 Computation of Epipolar Lines

Given a point $\mathbf{I}_1$ in one image, its homologous point $\mathbf{I}_2$ is constrained to lie within a line in the second image called the epipolar line of $\mathbf{I}_1$ (see figure 34). This line is simply the image by the second camera of the line $\mathbf{I}_1\,\mathbf{C}_1$, the inverse image by camera 1 of $\mathbf{I}_1$. It can also be seen as the intersection of the epipolar plane $\mathbf{I}_1\mathbf{C}_1\mathbf{C}_2$ with the image plane of the second camera.

For the sake of efficiency, it is important to be able to compute easily the epipolar line attached to any image point. First, it is obvious that any epipolar line in image 2 will go through a common point $\mathbf{E}_2$ called the epipole 2. $\mathbf{E}_2$ is simply the image by the second camera of the optical center of the first camera. Knowing the perspective transformation $\mathbf{T}_1$ of camera 1, the coordinates $[x_1, y_1, z_1, 1]^t$ of $\mathbf{C}_1$ are obtained by resolving:

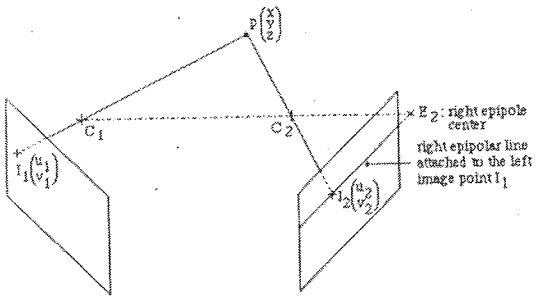


Fig. 34. Geometry of the epipolar lines in the right image.

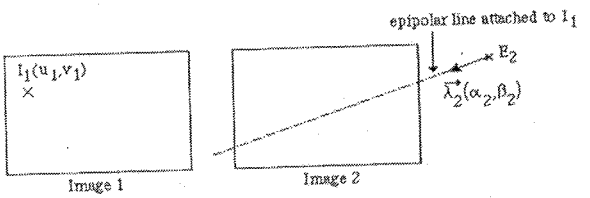


Fig. 35. Computation of the epipolar line attached to a point.

$$\begin{pmatrix}0\\0\\0\end{pmatrix} = \mathbf{T}_1 \begin{pmatrix}x_1\\y_1\\z_1\\1\end{pmatrix} \tag{2}$$

Then the image coordinates $[u_{e2}, v_{e2}]$ of $\mathbf{E}_2$ are obtained by applying $\mathbf{T}_2$ to $\mathbf{C}_1$:

$$\begin{pmatrix}su_{e2}\\sv_{e2}\\s\end{pmatrix} = \mathbf{T}_2 \begin{pmatrix}x_1\\y_1\\z_1\\1\end{pmatrix} \tag{3}$$

Given the position of the epipole, to determine the equation of an epipolar line, it is sufficient to compute the coordinates of a supporting vector λ_2 (see figure 35).

The remarkable point is that there is a *constant* linear relationship between the coordinates $(u_1, v_1, 1)$ of an image point $\mathbf{I}_1$ and the coordinates (α_2, β_2) of the supporting vector λ_2 of the corresponding epipolar line:

$$\begin{pmatrix}\alpha_2\\\beta_2\end{pmatrix} = \mathbf{M}_{12} \begin{pmatrix}u_1\\v_1\\1\end{pmatrix} \tag{4}$$

where $\mathbf{M}_{12}$ is a 2×3 matrix dependent on $\mathbf{T}_1$ and $\mathbf{T}_2$ exclusively and independent of the position of $\mathbf{I}_1$ (see Faugeras and Toscani [33] for details).

Of course the same formalism applies when inverting the roles of cameras 1 and 2. The 2×6 coefficients determining matrices $\mathbf{M}_{12}$ and $\mathbf{M}_{21}$ and the 2×2 coordinates of the epipoles $\mathbf{E}_1$ and $\mathbf{E}_2$ are computed once for all and stored at the end of the calibration procedure. During the matching procedure, these coefficients are used to compute in a straightforward manner the epipolar line attached to any point either in camera 1 or in camera 2.

8.3 Reconstruction of 3D Segments

When two segments S_1 and S_2 have been matched between the left and right images, it is possible to reconstruct a 3D scene segment S of maximum length whose images S_1' and S_2' are strictly included within S_1 and S_2 (see figure 36). This is done by first computing the epipolar lines attached to the vertices of each segment and by computing the corresponding intersection with the other segment. If there is an intersection, the vertex and the intersection point are kept as being the homologous images of a vertex of S. When two such pairs of homologous image points have been found, the three coordinates (x, y, z) of the vertices of S are computed as follows. For a given pair $(\mathbf{I}_1, \mathbf{I}_2)$ of homologous points, the following relations hold:

$$\begin{pmatrix}su_1\\sv_1\\s\end{pmatrix} = \mathbf{T}_1 \begin{pmatrix}x\\y\\z\\1\end{pmatrix} \tag{5}$$

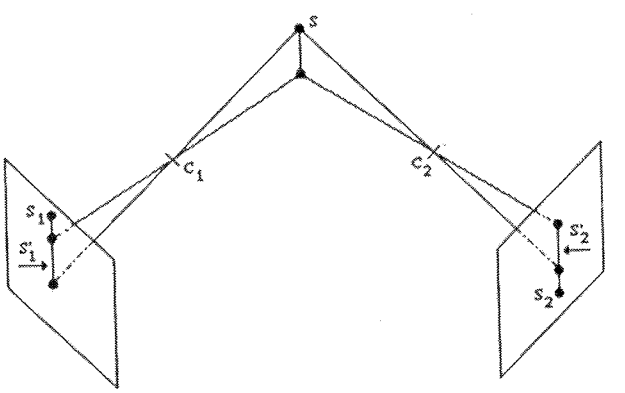


Fig. 36. Reconstruction of a scene segment from its two images.

$$\begin{pmatrix} s'u_2 \\ sv_2 \\ s \end{pmatrix} = \mathbf{T}_2 \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (6)$$

By eliminating s and s' in equations 5 and 6, one ends up with four linear equations of the three unknowns (x, y, z). In fact only three of these equations are linearly independent, because $\mathbf{I}_1$ and $\mathbf{I}_2$ have been chosen within a common epipolar plane. Therefore this system has an exact solution (x, y, z) straightforwardly computed.

9 Appendix B: Computation of the Disparity-gradient Limit

9.1 Definition

This definition is given for left-to-right matches. A symmetric definition is used for right-to-left matches. Given a left-to-right match (L, R, DISP) corresponding to a 3D point **P** (see figure 37), we want to relate the variation of depth of **P** with the variation of disparity between L and R. To do this, depth is computed along the line of sight of the left camera. A maximum variation of depth of ε around **P** corresponds to the points $\mathbf{P}_1$ and $\mathbf{P}_2$ whose images in the right image are $\mathbf{R}_1$ and $\mathbf{R}_2$, respectively, having disparities DISP + $\Delta 1$ and DISP + $\Delta 2$ with L. The idea is to connect a neighboring match (L', R', DISP') to (L, R, DISP) in the disparity graph if and only if the disparity gradient DISP'−DISP lies within the interval $[\Delta 1, \Delta 2]$.

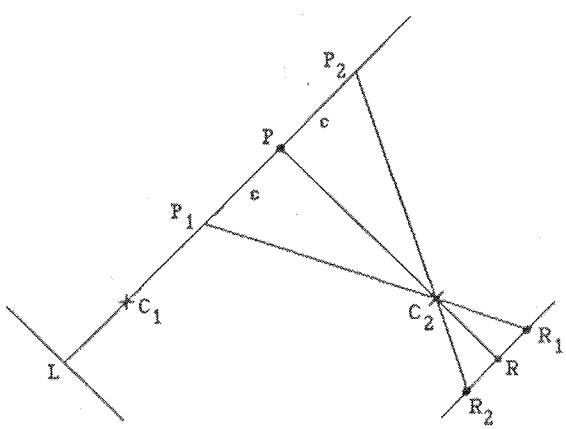


Fig. 37. Computation of the disparity gradient limit as a function of the point position.

9.2 Practical Computation

Given (L, R), we compute the coordinates of **P** by the technique described in section 8.3. Then, having the coordinates of $\mathbf{C}_1$, we compute

$$\mu = \mathbf{C}_1\mathbf{P}/\|\mathbf{C}_1\mathbf{P}\|$$
$$\mathbf{OP}_2 = \mathbf{OP} + \varepsilon\mu$$
$$\mathbf{OP}_1 = \mathbf{OP} - \varepsilon\mu$$

Finally the images of $\mathbf{P}_1$ and $\mathbf{P}_2$ are obtained by applying the perspective tranformation of the right camera to $\mathbf{P}_1$ and $\mathbf{P}_2$.

To speed up the computation of $\Delta 1$ and $\Delta 2$ during the propagation process, we build a lookup table during a preliminary off-line process. We choose a point L located in the middle of each bucket in the left image: then for each such point we determine n regularly spaced potential right matches on the corresponding right epipolar line lying within the a priori potential disparity range. For each pair (L, Ri, DISP_i) the values of $\Delta 1$ and $\Delta 2$ are computed and stored. During the matching process and given a current match (L, R, DISP), the program selects the bucket corresponding to L, and selects the disparities DISP_i and DISP_j closest to DISP. The tolerance values $\Delta 1$ and $\Delta 2$ are obtained by a simple linear interpolation. In the experiments with the indoor scenes, the values of $\Delta 1$ and $\Delta 2$ vary from a fraction of a pixel at a distance of 10 m to 5 or 6 pixels at a distance of 1.5 m (the value of ε was set equal to 20 cm).

Acknowledgments

The authors want to thank O.D. Faugeras for many helpful suggestions and comments throughout the course of this work, including among others the use of bucketing techniques. They also thank G. Toscani, who developed the calibration system, G. Giraudon for his work on the extraction of edge chains, and N. Rocher for the preparation of this report. Special thanks are due to F. Lustman for his substantial help in the developmental and experimental stages of this work. This work was partially supported by ESPRIT project P940.

References

1. D. Marr and T. Poggio, "Cooperative computation of

stereo disparity," SCIENCE vol. 194, pp. 283–287, 1976.
2. W.E.L. Grimson, FROM IMAGES TO SURFACES. M.I.T. Press: Cambridge, MA, 1981.
3. W.E.L. Grimson, "Computational experiments with a feature based stereo algorithm," M.I.T., Cambridge, MA, A.I. Memo. 762, 1984.
4. Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search," IEEE TRANS. PAMI vol. PAMI-7, pp. 139–154, 1985.
5. H. Baker and T.O. Binford, "Depth from edge and intensity based stereo," in PROC. 7TH INT. JOINT CONF. ARTIF. INTELL., Vancouver, Canada, 1981, pp. 631–636.
6. K. Ben Rhouma et al., "A K2-D perception approach for assembly robots," in PROC. EURASIP 1983, Erlangen, FRG, 1983.
7. M. Benard, "Extracting 3-D coordinates of an object from a digital stereopair: an automated method," in PROC EURASIP, Erlangen, FRG, 1983, pp. 227–230.
8. R. Mohr and W. Wrobel, "La correspondance en stéréovision vue comme une recherche de chemin optimal" [in French], in PROC. 4TH AFCET–INRIA CONF. REC. FORMES INTELL. ARTIF., 1984, pp. 71–79.
9. G. Medioni and R. Nevatia, "Segment based stereo matching," in PROC. IMAGE UNDERSTANDING WORKSHOP, Arlington, VA, 1983, pp. 128–136.
10. M. Berthod and P. Long, "Graph matching by parallel optimization methods: an application to stereo vision," in PROC. 7TH INT. CONF. PATTERN RECOGNITION, Montreal, Canada, 1984, pp. 841–843.
11. S.T. Barnard and W.B. Thompson, "Disparity analysis of images," IEEE TRANS. PAMI vol. PAMI-2, pp. 333–340, 1980.
12. J.E.W. Mayhew and J.P. Frisby, "Computational and psychophysical studies towards a theory of human stereopsis," ANTIF. INTELL. vol. 17, pp. 349–386, 1981.
13. S. Pollard, J. Porril, J. Mayhew, and J. Frisby, "Disparity gradient, Lipschitz continuity, and computing binocular correspondence," University of Sheffield, England, Intern. Rep. AIVRU-010, 1986.
14. H.K. Nishihara, "PRISM, a practical real-time imaging stereo matcher," M.I.T., Cambridge, MA, A.I. Memo. 780, 1984, 31 pages.
15. S. Castan and J. Shen, "A stereo vision algorithm taking into account the perspective distorsions," in PROC. 7TH INT. CONF. PATTERN RECOGNITION, Montreal, Canada, 1984, pp. 444–446.
16. H.P. Moravec, "Towards automatic visual obstacle avoidance," in PROC. 5TH INT. JOINT CONF. ARTIF. INTELL., Cambridge, MA, 1977, p. 584.
17. R. Nevatia, "Depth measurement by motion stereo," COMPUT. VISION, GRAPHICS AND IMAGE PROCESSING vol. 5, pp. 203–214, 1976.
18. D.B. Gennery, "A stereovision system for an autonomous vehicle," in PROC. 5TH INT. JOINT CONF. ARTIF. INTELL., Cambridge, MA, 1977, pp. 576–582.
19. K. Mori, M. Kidode, and H. Asada, "An iterative prediction and correction method for automatic stereocomparison," COMPUT. VISION, GRAPHICS IMAGE PROCESSING vol. 2, pp. 393–401, 1973.
20. M.D. Levine et al., "Computer determination of depth maps," COMPUT. VISION, GRAPHICS IMAGE PROCESSING vol. 2, pp. 131–150, 1973.
21. J.J. Hwang and E.L. Hall, "Matching of featured objects using relational tables from stereo images," COMPUT. VISION, GRAPHICS IMAGE PROCESSING vol. 20, pp. 22–42, 1982.
22. G. Poggio and T. Poggio, "The analysis of stereopsis," ANNU. REV. NEUROSCI. vol. 7, pp. 379–412, 1984.
23. N. Ayache and B. Faverjon, "Fast stereo matching of edge segments using prediction and verification of hypotheses," in PROC. CONF. COMPUT. VISION AND PATTERN RECOGNITION, San Francisco, CA, 1985, pp. 662–664.
24. N. Ayache and B. Faverjon, "A fast stereovision matcher based on prediction and recursive verification of hypotheses," in PROC. 3RD WORKSHOP COMPUT. VISION: REPRESENTATION AND CONTROL, Bellaire, MI, 1985, pp. 27–37.
25. D. Marr and E. Hildreth, "Theory of edge detection," PROC. R. SOC. LONDON [B] vol. 207, pp. 187–217, 1980.
26. N. Keskes, A. Boulanouar, and O.D. Faugeras, "Application of images analysis techniques to seismic data," in PROC. INT. CONF. ASSP, Paris, pp. 855–857, 1982.
27. N. Ayache and O.D. Faugeras, "Hyper: a new approach for the recognition and positioning of 2-D objects," IEEE TRANS. PAMI vol. PAMI-8, pp. 44–54, 1986.
28. J.F. Canny, "Finding edges and lines in images," M.I.T. Artif. Intell. Lab., Cambridge, MA, A.I. Memo. 720, 1983.
29. R.Deriche, "Optimal edge detection using recursive filtering," INT. J. COMPUT. VISION, 1987.
30. T. Pavlidis, STRUCTURAL PATTERN RECOGNITION. Springer-Verlag: New York, 1977.
31. M. Berthod, "Polygonal approximation of edge chains," INRIA Intern. Rep. 1987 (in press).
32. D. Knuth, THE ART OF COMPUTER PROGRAMMING, vol. 3: SORTING AND SEARCHING. Addison Wesley: Reading, MA, 1975.
33. O.D. Faugeras and G. Toscani, "The calibration problem for stereo," in PROC. CONF. COMPUT. VISION AND PATTERN RECOGNITION, Miami, FL, 1986, pp. 15–20.
34. R.D. Arnold and T.O. Binford, "Geometric constraints in stereo vision," in PROC. SPIE, 238, San Diego, CA, 1980, pp. 281–292.
35. N. Ayache, O.D. Faugeras, B. Faverjon, and G. Toscani, "Matching depth maps obtained by passive stereovision," in PROC. 3RD WORKSHOP COMPUT. VISION: REPRESENTATION AND CONTROL, Bellaire, MI, 1985, pp. 197–204.
36. O.D. Faugeras, N. Ayache, and B. Faverjon, "Building visual maps by combining noisy stereo measurements," in PROC. INT. CONF. ROBOTICS AND AUTOMATION, vol. 3, San Francisco, CA, 1986, pp. 1433–1438.
37. N. Ayache and O.D. Faugeras, "Building, registrating and fusing noisy visual maps," in PROC. 1ST INT. CONF. COMPUT. VISION, London, England, 1987.