Stanford Artificial Intelligence Laboratory
Memo AIM-268

October 1975

Computer Science Department
Report No. STAN-CS-75-521

Depth Preception in Stereo Computer Vision

by

Clark Thompson

Research sponsored by

Advanced Research Projects Agency
ARPA Order No. 2494

COMPUTER SCIENCE DEPARTMENT
Stanford University

Stanford Artificial Intelligence Laboratory
Memo AIM-268

October 1975

Computer Science Department
Report No. STAN-CS-75-521

Depth Preception in Stereo Computer Vision

by

Clark Thompson

ABSTRACT

This report describes a stereo vision approch to depth perception; the author has build upon a set of programs that decompose the problem in the following way: 1) Production of a camera model: the position and orientation of the cameras in 3-space. 2) Generation of matching point-pairs: loci of corresponding features in the two pictures. 3) Computation of the point in 3-space for each point-pair. 4) Presentation of the resultant depth information.

The author's present addess is Computer Science Department, Carnegie Mellon University, Schenley Park, Pittsburgh, Pa 15213

This research was supported by the Advanced Research Projects Agency of the Department of Defense under Contract DAHC 15-73-C-0435 . The views and conclusions contained in this document are those of the author(s) and should not be interpreted as necessarily representing the official policies, either expressed or implied, of Stanford University, ARPA, or the U. S. Government.

Reproduced in the U.S.A. Available from the National Technical Information Service, Springfield, Virginia 22151.

I. Statement and History of the Problem.

The computer depth perception problem is the derivation of the distance from the camera to each point of the viewed scene.

Much work has been done with monocular views of the so-called "blocks world": white polyhedra on a dark table. Here the picture is first reduced to a line drawing, which the computer "perceives" in much the same way that a human might, applying task-specific prior knowledge to resolve ambiguities. If the location of the table surface is known in advance, and all objects are assumed to be supported by the table (or other blocks), the depth perception problem is solved. Gunnar Grape [Grape] gives a description of his and previous approaches to this rather limited task domain.

Other methods have been developed to derive depth information from a single camera. Berthold Horn [Horn] used shading clues; a group at Stanford [Nevatia] has investigated laser ranging. The former technique has only limited utility in a general environment, while the latter is quite successful for near-field work (a powerful laser is required to extend this range).

Multiple views of a single object were used by Bruce Baumgart [Baumgart] to derive a model of that object. He located the silhouette of a toy doll or horse as it rotated on a turntable, deriving its shape from the intersection of the "silhouette cones". This is an inefficient way of deriving depth information, in the following sense: in each picture, only the silhouette of the object is located in 3-space for each picture, whereas it is possible to do much more than that (vide this report). Also, this method presupposes an ability to determine the object-background boundary, a nontrivial problem in itself under normal lighting conditions.

This report describes a stereo vision approach to depth perception; the author has built upon a set of programs that decompose the problem in the following way:

1. Production of a camera model: the position and orientation of the cameras in 3-space.
2. Generation of matching point-pairs: loci of corresponding features in the two pictures.
3. Computation of the point in 3-space for each point-pair.
4. Presentation of the resultant depth information.

Sub-problem 1 has been adequately solved [Hannah], for high-quality picture pairs with relatively small (<10%) perspective distortion of corresponding objects.

Hannah also attacked sub-problem 2; the present report describes several refinements on her approach to make it less sensitive to perspective distortion, less dependent on human interaction, and a little better at accepting or rejecting prospective point-pairs. A group at JPL [Levine] handles this sub-problem in a quite different way: the methods are compared in section III.

Sub-problem 3 is a trivial exercise in trigonometry and linear algebra, given the camera model generated by sub-problem 1.

This report describes a program that takes the points in 3-space and produces output intended for human consumption--a GEOMED source file (GEOMED, a "geometrical editor", allows examination of three-dimensional line drawings). A naive attempt to discriminate objects within the scene is incorporated in this program.

II. Conventions and Basic Concepts of Stereo Picture Processing.

A picture is a two-dimensional array of integer values which represent the light intensities of a scene, as seen through some camera, at a set of sample points. Several parameters are of immediate interest, such as the imaging geometry of the camera, the number and spacing of the sample points (spatial resolution), and the precision to which the light intensities are recorded (gray-scale resolution). In this report, all cameras have relatively long focal lengths, so that "pin-cushion" distortion is minimal (pin-cushioning is the distortion produced by a "fish-eye" lens). Due to space limitations in the computer, spatial resolution is limited to some tens of thousands of sample points per picture, while gray-scale resolution is normally 6 bits (64 intensity levels). This leads to an image quality quite similar to that of a normal television picture.

Sample points ("pixels") fall on a rectangular grid; Cartesian coordinates are the natural choice. In keeping with the conventions used in the television industry, pixels are identified by their (I,J) positions with respect to the upper left-hand corner of the picture, which has position (0,0). The I-dimension increases to the right; the J-dimension increases downward. To distinguish between pixels in the two pictures comprising a stereo pair, (IA,JA) are used as coordinate labels in one picture, (IB,JB) in the other.

Stereo matching is the process of finding areas in the two pictures that correspond to the same 3-D piece of scenery in the "real world". For example, the area around (130,115) in the left picture of the barn pair (see Section IX) matches the area around (15,115) in the right picture: both are views of the fence post in the foreground. Intuitively, one area matches another if the intensity values of corresponding pixels are nearly equal. Exact pixel-by-pixel equality is never observed, due to "errors" from a number of sources. First, the cameras are looking at this piece of scenery from different points of view, thus changing its apparent shape and shading. Second, potential matching areas in the two pictures must be centered on actual sample points, since these are the only places that intensities have been observed: interpolation of intensity values between pixels is slow and inaccurate. Thus a "matching area" is merely within a pixel of the correct match, and the observed intensity values at the approximately corresponding pixels are not expected to be equal. Finally, the cameras are far from perfect: differences in "gain", "offset", and "noise" are to be expected. A statistical method of detecting approximate matches is clearly indicated; normalized correlation has been chosen as the match metric, for a variety of reasons (see [Hannah]). This gives a "score" between -1 and +1 for the closeness of match between two areas, where +1 is attainable only by the perfect match (two areas only differing in relative gain and offset).

The term "matching point-pairs" is shorthand for "the pair of points that lie at the center of a pair of matching areas". For computational reasons, these areas are rectangular windows.

Stereo matching is not an infallible means of analysis. "False matches" are an ever-present problem arising for two reasons. There may be multiple, highly correlating matches caused by repetition of features: imagine trying to match two views of a freshly-painted picket fence against a uniform background. Also, the discovery of a correct match may be blocked by the occlusion of the scenery seen in one picture by a closer object; a spuriously high correlating point-pair may be selected instead.

More reliable matches can be obtained by increasing the size of the correlation windows, thus producing more significant correlations (in the statistical sense). Unfortunately, large windows make it impossible to match near the edges of objects: the windows will include non-matching background along with the matching object. Also, large windows are more sensitive to perspective distortion. Windows containing 121 pixels seem to be a good compromise.

III. Generation of Matching Point-Pairs.

The approach described herein is due to Marsha Jo Hannah [Hannah], who in turn relied on the image processing groundwork laid at Stanford AI by Lynn Quam. Briefly, the technique used is the examination of a subset of all possible point-pairs in the two pictures, making the decision "match" or "no match" for each. For computational speed and ease, many "candidate" areas in picture B are compared against a single "target" area in picture A until a match is found.

An exhaustive examination of all point-pairs is clearly out of the question, when the pictures under consideration contain tens of thousands of sample points. Thus, various search-reduction techniques have been developed. The most important of these is the continuity assumption: if (IB,JB) matches (IA,JA) then the match for (IA+DI,JA+DJ) should be found close to (IB+DI,JB+DJ); and if (IB,JB) is a very poor match for (IA,JA) then there is not much hope of finding a good match in the vicinity of (IB,JB). Finally, given a camera model, trigonometrically possible matches for (IA,JA) will be found on a straight line in picture B. This is the "matching line" derived in [Hannah].

At the heart of the point-pair generator is a "region grower", which tries to find matches for the four nearest neighbors of every matched "target" area. It uses the continuity assumption to compute the most likely "candidate" area for a given neighbor: if this is not an acceptable match, then a local search is initiated. All matches found by examining the neighbors of a given "seed" point-pair are said to belong to a "region". A "region" is thus a portion of the scene with no depth discontinuities.

An alternative approach, explored by the JPL group [Levine], dispenses with the overhead of remembering the perimeter of the current region by picking target areas in a uniform top-to-bottom, row-by-row manner. The point-pairs of the preceding row are used to compute likely candidate areas for the current row. It is not clear to the author which approach is the more efficient.

IV. Criteria for Accepting a Match.

The decision between "match" and "no match" is the most interesting part of point-pair matching. Many heuristic techniques have been proposed; the author's program uses the following:

1. Calculation of the variance of the target area, rejecting it if below a threshold value. This avoids making a "match" on the basis of insufficient information--all pieces of clear sky or blank wall are indistinguishable in the pictures.
2. Threshold rejection of the target area on the basis of its "directed variance", or the ratio of information content perpendicular to and parallel to the baseline direction. This is also justified on an informational argument: candidate areas will be selected along the "matching line" approximately parallel to the baseline, so that information is needed along the baseline direction to discriminate between adjacent target areas.
3. A local search in the vicinity of a high correlation match, to find a correlation maximum.
4. Comparison of the correlation maximum with the "auto-correlation" of the target window--the average correlation of the target window with itself when shifted by one pixel in each direction. This attempts to measure how difficult it is to match the target window: if it has a lot of "high frequency" information, the correlation peak will be much sharper and its maximum ascertained much less precisely (much more conservatively).

The variance thresholding tests are lifted from Hannah's program. I have not subjected these to any tests of validity, although they seem to do the right thing.

The local search for a correlation maximum is intuitively justifiable; even if a particular match passes all significance tests, it is less likely to be correct than a neighbor with a higher correlation.

The auto-correlation test is the result of my experience with correlation matching. If the highest correlation is plotted against the auto-correlation for each matchable target area, an interesting pattern emerges: the average correlation is just the average of 1.0 and the auto-correlation (see section X for a histogram of this relation). This may be understood as the result of an average error of half a pixel for each match. Also, the probability of finding a correlation maximum less than the auto-correlation is seen to be quite small. This indicates an empirical threshold value for accepting a correlation:

$$\text{THRESHOLD} = K + (1-K)*\text{AUTOCORRELATION}.$$

The value of K can be varied to make the threshold more or less strict: K=0 screens out only extremely unlikely correlations, while K=.5 will disallow half of the good ones. The former is appropriate when making a global search for a match, the object being to avoid making a mis-match while still having a good probability (.5) of finding a match. The latter is used to evaluate the results of a local search for a match to a target immediately adjacent to a previously obtained "good" match.

A few "bad" matches were obtained with this threshold function; all had very low auto-correlations, and their correlation was less than .5. The obvious fix was applied:

$$\text{THRESHOLD} = \text{MAX}(K1+(1-K1)*\text{AUTOCORR} , K2).$$

This is the correlation significance test used in the author's program.

V. Perspective Distortion.

The techniques described above for stereo matching using fixed-size windows work very well when objects appear the same size and shape in both pictures. This ideal condition will only be observed when the surface of the object being viewed is far from both cameras (relative to the baseline distance), and more or less parallel to the baseline. A glance at the "barn" and "yard" pairs (section IX) will show that one should expect significant deviations from ideality. The face of the barn appears nearly half again as large in the right hand "barn" picture as in the left; the log in center left of the "yard" pair changes shape dramatically from one picture to the other. Perspective distortion is definitely non-linear globally; locally, it can be well represented as a linear function. Two types of corrections for distortion have been implemented: "uniform scaling" and "directional scaling".

Uniform scaling attempts to account for distortion due to the different distances of the matched piece of scene from the two cameras. If an object is twice as close to camera A than to camera B, it will appear twice as big in picture A. The proper correction for this effect is obvious: one merely makes the target window twice as small in each dimension as the candidate window. The relative distance, and hence the scaling factor, is easily computed from the location of the proposed target window, using the camera model. However, the operation of scaling the target window to an arbitrary size is more difficult than it sounds. If the candidate window is 11*11 pixels, the target window should be 5.5*5.5 pixels in size, while 121 sample points are still required from each window to do the correlation calculation. Unfortunately, sample points lie only on integral coordinates; some form of interpolation is required. To avoid slowing down the correlation calculation inordinately, the closest pixel is used as the "interpolated" value.

Due to the nature of the test pictures available at this time, the uniform scaling correction for perspective distortion has not been adequately tested: the largest distance ratio found in the "barn" and "yard" pictures is 1.06 (for the fence-post in the "barn" picture).

Directional scaling corrects for the distortion due to different orientations of the face of an object relative to the viewing angles of the cameras. Consider the plane formed by the lines from each camera center througn the center of its "matching area" (these lines are normally skewed, but nearly intersect at location of the viewed object in 3-space; redraw the lines so that they intersect midway between the nearest approach of the original lines). A small portion of the face of an object may be approximated by a square whose center is at the line intersection, and whose orientation is described by two angles. One angle is the dihedral between the plane and the square. The ground is normally at a small dihedral angle, while objects such as trees and barns have dihedral angles of around 90 degrees. The second angle is that between the projection of the surface normal of the square onto the plane and the line connecting a camera with the line intersection. In general, this "normal" angle is different when measured with respect to each camera. For example, the door of the barn (in the "barn" pair) is inclined at about 83.5 degrees to the viewing angle of camera A, but only 80 degrees to camera B.

The directional scaling implemented in the author's program accounts for the relative size changes due to differences in the normal angles. The ratio of the cosines of these angles gives the apparent size ratio (1.6 for the barn door). The apparent size ratio differs most from unity when the viewing angle is large; only then will a small change in normal angle give a large ratio of cosines. The magnitude of the change in normal angle is inversely proportional to the distance from camera to object, by simple geometry. Thus directional distortion is most important in the "near field". A correction for directional scaling is implemented by varying the aspect ratio of the target window: on the barn door, an 11*11 candidate window is best matched by a 11*17 target window. As in uniform scaling, the problem of non-integral coordinates for sample points arises, and the same solution was adopted: the value of the closest pixel is used. Only integral target window sizes are allowed, to avoid redundant comparisons.

The magnitude of the correction for directional scaling cannot be predicted without knowing the angle of inclination of the face of the object under consideration, so that a "search" must be done for each prospective point-pair to find the best correction factor. The time spent in this search is reduced by the application of two insights. First, by "continuity", the directional distortion for a point-pair a point-pair should be similar to that of neighboring point-pairs. Second, if an upper limit is placed on the magnitude of normal angles (typically 85 degrees), then the number of different aspect ratios to be

tried for the target window is very small (3) for point-pairs corresponding to far-away objects (100 baselines). This form of directional scaling correction is moderately successful in increasing correlation scores for point-pairs corresponding to objects with large normal angles: without it, less than half of the door of the barn can be matched (when thresholds are adjusted to nearly eliminate "false matches"), while nearly all is matchable using directional scaling. Knowledge of normal angles should be useful after the matching process, both in discarding point-pairs that don't agree with their neighbors, and for modeling of the 3-D scene (neither of these schemes have been investigated at the present time).

Correcting for distortion due to dihedral angle could be implemented at a reasonable cost in running time: a square candidate window should be matched to a parallelogram target. This would most assuredly help in correlating along the ground plane--in fact, moderate success could probably be acheived by trying only a few dihedral angles for each prospective point-pair. In the barn pair, a 11*11 window of the grassy field in the middle of the left picture would be best matched in the right picture by a parallelogram whose top edge is skewed two pixels to the right of the bottom edge. Correction for dihedral angle distortion is, in the author's view, the most promising area for future research in stereo matching.

VI. 3-D Modeling.

In testing and debugging the point-pair matching routines described in this report, the author soon felt the need for a convenient means of viewing the resultant depth information. A program already existed (GEOMED) that was capable of displaying the appearance of three-dimensional line drawings from any orientation; the generation of such drawings seemed a trivial matter. The obvious approach was to draw lines between all the 3-space points that correspond to neighboring target areas in picture A. Unfortunately, this produced far too many lines for GEOMED to handle effectively. Relatively unimportant lines can be removed by the following rules: where a pair of lines cross, delete one of them; where four non-collinear points are approximately co-planar, delete "diagonal" or interior lines; and where a line connects two "objects", delete it. Of course, the discrimination of objects is a non-trivial matter, and is not handled very successfully by the simple algorithm described below.

First some basic observations. Each point-pair produced by the matching process corresponds to a unique point in three-space (X,Y,Z). An estimate of the error in the Z coordinate can be computed by generating the (X',Y',Z') corresponding to a point-pair with one pixel error in the baseline (I) direction: (error in Z) = |Z - Z'|. An "object" may be thought of as a group of adjacent points with approximately the same value of Z: some multiple of the computed error in Z may be used to give quantitative meaning to the word "approximately". Similarly, a "face" of an object is an approximately planar collection of points in the object. A reasonable GEOMED drawing can consist of merely the perimeters of all faces.

Algorithm for object discrimination:

Sort all 3-D points by their (IA,JA) coordinates, and draw lines between all points that are adjacent in picture A (horizontally, vertically, and diagonally). Assign the upper left-hand matched point of picture A to object #1, then try to extend the size of the object by including more and more points (moving only along the lines connecting adjacent points). As points accumulate in an object, eventually a surface plane will be determined: components of the change in Z with respect to changes in IA and JA coordinates can then be calculated. Hereafter, the entrance requirement for points is that their Z coordinate match the extrapolated Z coordinate of the surface plane, within a wide error bound (typically 50 times the estimated error in the Z coordinate). When no more points can be included in object #1, start object #2 with the upper-left-most point that hasn't already been included in an object. Continue in this manner until all points have been assigned an object, then delete lines connecting points in different objects.

Algorithm for face discrimination:

First, all possible non-overlapping triangular faces must be constructed, whose three vertices lie in the same object. Then adjacent triangles may be assigned to "faces", starting with the upper-left-most triangle. Here the components of dZ/dIA and dZ/dJA are determined by the first triangle assigned to a "face"; succeeding triangles must share an edge and hence two vertices with the "face", and the Z coordinate of the new vertex must agree with the extrapolated Z coordinate within a narrow error bound (typically twice the estimated error in its Z coordinate).

The line drawing of the "stump" (see section VIII) was processed using the above algorithms. There were 627 vertices and 2179 lines in the "raw" drawing, before lines interior to faces and those connecting objects were removed; after processing, only 402 lines remained. The matching points were partitioned into three major objects: the gate post in the background, the rear-most portion of the axe handle, and the stump, including the protruding axe handle and some of the ground on either side of the stump. Clearly, this object discrimination algorithm needs much refinement.

In the future, better object and face discrimination may form part of a "feedback loop" that discovers bad point-pairs, predicts the appearance of the scene from different viewpoints, and "zooms in" on interesting features (calls for high-resolution matching over limited regions such as the stump in the "yard" scene).

VII. Program Description.

The author's stereo vision program is divided into three units: a point-pair matcher, a point-pair analyzer, and a 3-D modeler.

The point-pair matcher is named ZPGROW. It asks the operator for a Data Disk overlay channel (for visual output of point-pairs as they are created), and for the name of a pair of pictures and their associated camera model. At this point, a number of debugging routines are available. Typing "RP" calls the automatic point-pair matcher; the program prompts for parameters, i.e., boundaries and grid spacing for the prospective target areas in Picture A. Point-pairs are generated in the following manner:

1. (Outer loop) All unmatched points in picture A lying within the entered boundary are given to step 2. Points near the center of picture A are tried first. Points with unacceptable variance are discarded immediately.
2. (Global match) The "matching line" in picture B is calculated for the given point in picture A, then every third point along this line is used as the center of a candidate area. No search is made for directional scaling factors. All correlations above a threshold are sorted; the three highest scoring point-pairs are passed to step 3.
3. (Hill-climb) A three-dimensional hill-climb is performed, varying IB,JB, and directional scaling factor to optimize the correlation score. The (IB,JB) coordinates are constrained to lie within a certain distance from the computed "matching line" (typically 1 pixel). The directional scaling factor is bounded as described in section IV. If the optimum correlation is above the threshold, the corresponding point-pair is passed to step 4.
4. (Region queue) The target and candidate areas are marked GOOD, and the point-pair is saved on a disk file. Four neighboring point-pairs are extrapolated from the parameters of the matching one, and entered on a FIFO queue. Start step 5.
5. (Region grower) If the region queue is empty, go to step 6. Otherwise, a point-pair is taken off the queue and examined. If it has not been matched, and its variance is OK, its correlation is checked against the correlations of 4 neighboring target areas and 2 different directional scaling factors. If it is not a local maximum, the target area that had a higher correlation is checked in the same manner; this "hill-climbing" is bounded by the same constraints as in step 3. If a correlation peak is found that exceeds a threshold, that point-pair is given to step 4. If no correlation peak is found, the point-pair is put on the mis-match queue (this is only done once for any target area). If a sub-threshold peak is found, it is so marked; it will not be checked again in this run of the region grower. Continue step 5.
6. (Mis-match retry) If the mis-match queue is empty, continue the outer loop. Otherwise, take a point-pair off the mis-match queue and give it to step 3.

The lack of finesse in the "outer loop" should be obvious: this brute force technique, while slow, results in nearly as complete a matching as possible. The inner loops gain much speed from an elaborate "marking" scheme, which places target areas in one of 8 states: virgin, matched, bad variance, etc.; this reduces redundant computation. The basic structure of the program (except for the outer loop) is due to Hannah. The author merely "patched in" perspective distortion corrections, a new correlation threshold, and the outer loop.

Correlations are calculated in a "direct" way, that is, computing the squared sum of the errors (a few computational tricks greatly speed this process). As is commonly known, the FFT may be used to compute correlations in N*log(N) time, as opposed to the N*N time for the direct method; however, in this case N is only 121, so that the tradeoff point between increased overhead and poorer asymptotic behavior has not been reached for this computer (a PDP-10) [Hannah]. Although great care was taken in coding the correlation calculation, it still accounts for most of the running time of the program. Levine [Levine] reports the use of relatively inexpensive hardware that speeds this calculation by a factor of ten or more; still further increases in speed would be possible with specially-designed hardware assistance.

The point-pair analyzer, MANNA, takes the disk file of point-pairs created by the automatic point-pair generator, sorts them by IA or JA coordinates, and creates: 1) a listing file of the

point-pairs, and/or 2) a file suitable for use by the 3-D modeler.

The 3-D modeler, MKB3D, makes a .B3D file suitable for viewing with GEOMED. The operator must enter parameters for object and face discrimination, as described in section V. GEOMED is not completely happy with this B3D file, as it expects convex polyhedra instead of "lamina." However, the display routines (with the exception of the hidden line eliminator) work perfectly well.

VIII. Conclusions.

This project has demonstrated the feasibility of perspective distortion correction in computer stereo vision. Much more work needs to be done in this area, including:

1. Correction for dihedral distortion, as defined in section V, to facilitate matching along the ground plane.

2. Implementation of a termination test to avoid the fruitless global searching obtained in the "stump" pair matching (see section IX).

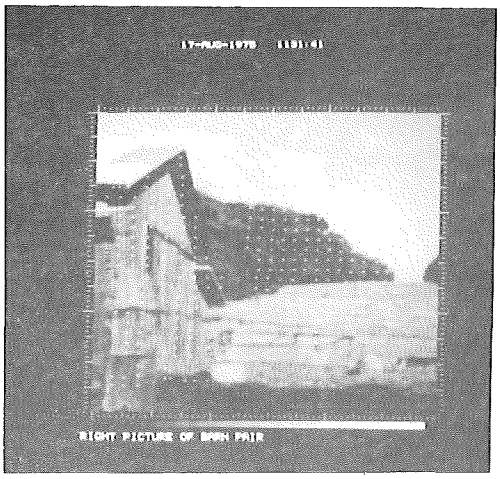
3. Refinements on the "face" and "object" discrimination algorithms of section VI.

4. Acquisition of special purpose hardware to speed up correlation calculations. The PDP-10 used for this research takes milliseconds to compute a correlation; order-of-magnitude improvements are quite possible with present-day technology.

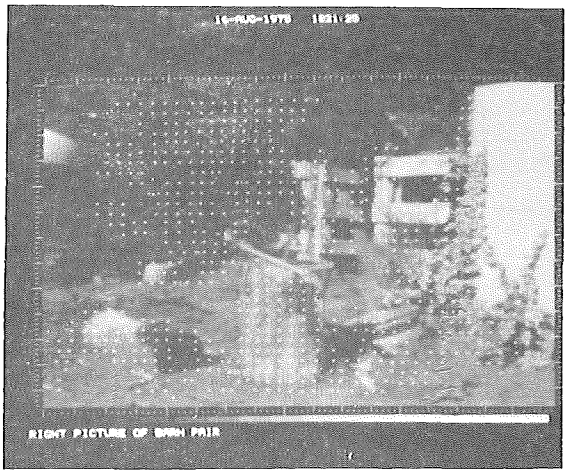
Even given all these improvements, "real time" computer depth perception lies far in the future, judging by the size of the gap between the best current efforts and the performance of humans.

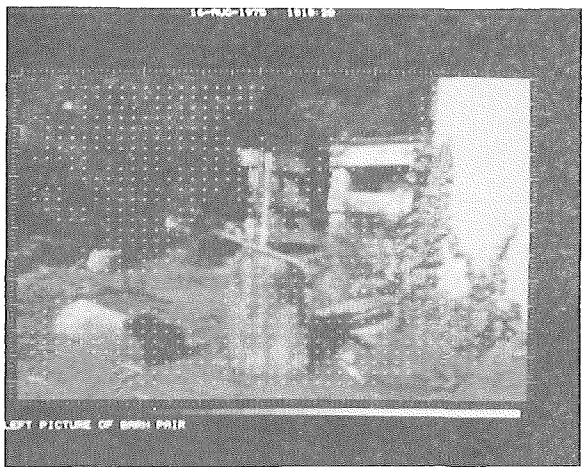
IX. Illustrations.

The illustrations below are snapshots of the monitor ("television") output produced by ZPGROW. Two sets of stereo pictures were used as input data: the "barn" and "yard" pair. These were digitized at high resolution, then spatially averaged to result in the reduced, full views of the yard and barn; the "stump" pair is just an enlarged version of the stump in the middle of the "yard" pair.

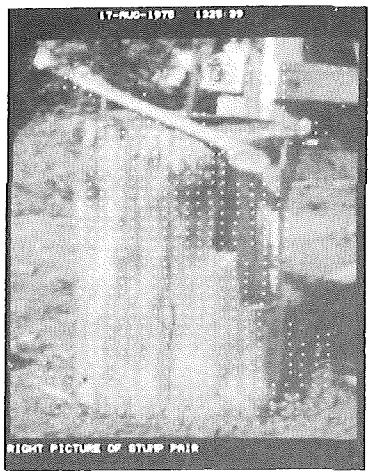


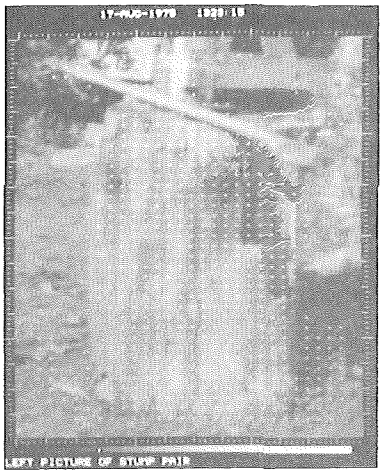
The result of a complete matching on the "barn" pair (on a 5*5 grid). The white spots are at the center of matching areas; 336 point-pairs were generated in 3 minutes of CPU time. Note that the wooded hillside to the right of the barn is correctly matched; there is a large region visible in the right-hand picture that is occluded by the barn in the left-hand one. The barbed-wire fence in extreme fore-ground presents an obstacle to the matching of areas of the grassy field in middle-ground. The horizontal spacing of the dots is much wider on the barn door in the right picture than the left, indicating a moderate amount of perspective distortion.





A complete matching on the "yard" pair, again on a 5*5 grid spacing; 774 point-pairs were generated in 8.5 minutes. Nearly all the point-pairs were obtained from a single global match. The increased time spent in finding each point-pair, relative to that observed for the "barn", is due to two factors: first, very little time was spent on the sky of the "barn", as its variance was too low; second, the "yard" pair is half again as wide as the "barn" pair, so that the global match (step 2 in section VII) took longer. Most of this pair is correctly matched, except for some low variance areas of dirt (in middle ground) and trees (in background). The two stumps in the foreground are the most interesting to match, due to large perspective distortion.

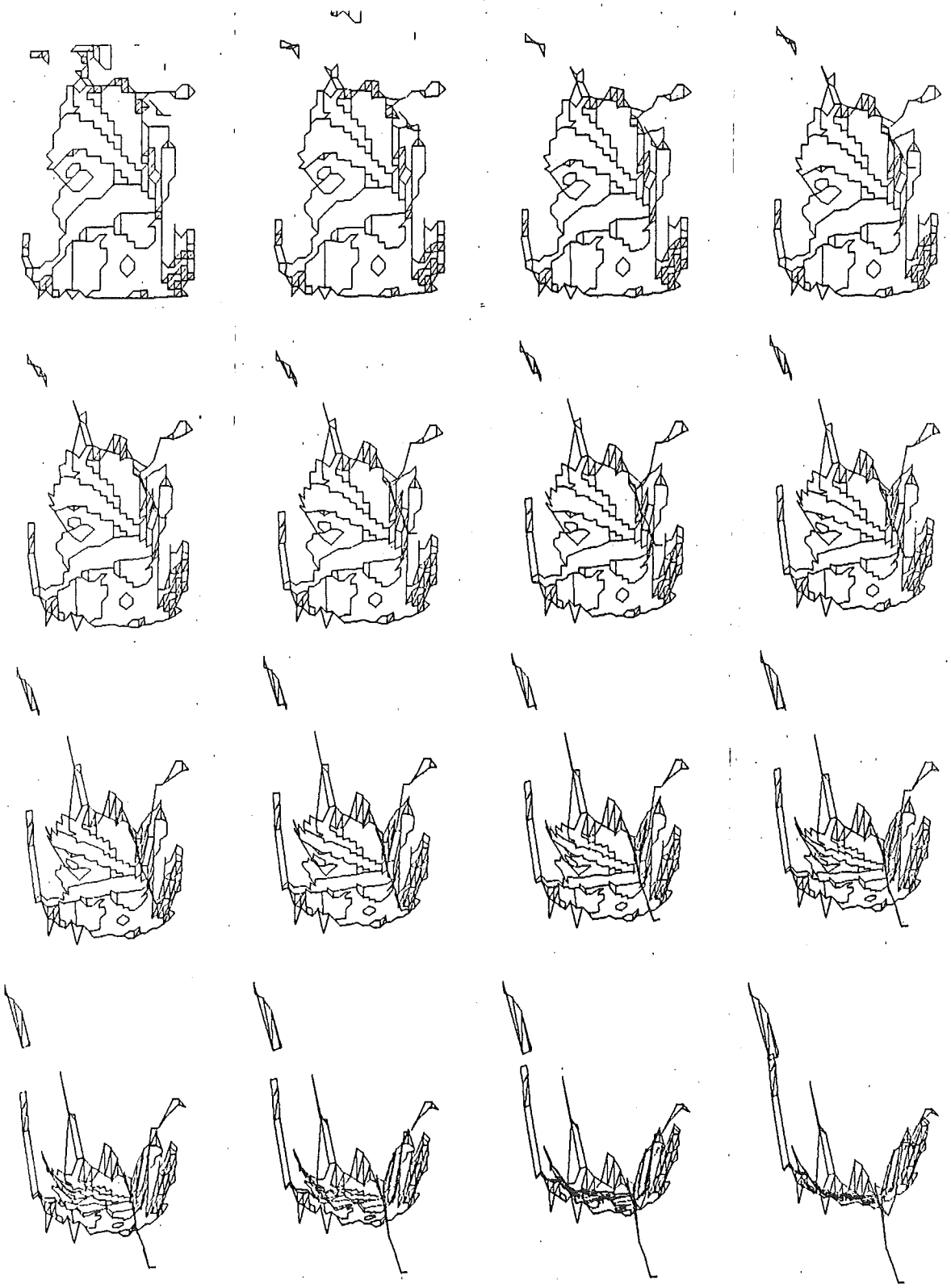




A complete matching of the "stump" pair (5*5 grid) produced 627 points in 13.5 minutes. Most of this time was spent in fruitless global searches (step 2 in section VII); 582 of these matches were found in the first 2.5 minutes. Clearly, a more intelligent termination algorithm would be helpful. A finer grid would have matched more of the axe handle on top of the stump; as it is, only a few points of it are located. A large amount of dihedral distortion prevents the accurate matching of more of the ground.

The results of the "stump" pair matching above were fed to MANNA and MKB3D; GEOMED was used to depict a line drawing of the stump as viewed from sixteen orientations, reproduced on the next page. At top left, the stump is seen "head on" from Camera A (see the left hand picture of the "stump" pair, above). Proceeding from left to right, top to bottom, the camera ia raised in an arc centered at the stump, so that in the bottom right picture the camera is directly above the stump. In this last picture, a portion of the axe handle is clearly seen to protrude toward the original camera position, or downwards from this view. In the upper left hand portion of each picture, a cluster of matching point-pairs at the far end of the axe handle remains nearly stationary throughout the rotation process, merely changing its shape and size from a small "bow tie" in the views at the top of the page to a large trapezoid in the bottom views. A portion of the ground to the right of the stump elongates vertically from the first to the last picture, eventually becoming partially occluded by the far end of the axe handle in the last picture.

Views of the Stump.



X. Correlation / Auto-correlation Histogram.

This histogram was prepared form the results of complete matchings of the "yard", "barn", and "stump" pairs, counting the total number of good matches found for a given range of auto-correlation and correlation values. For example, 55 matches of correlation .95 to 1.0 were found for windows of auto-correlation from .90 to .95.

As matches are deemed "good" and thus included in the histogram only if they pass the author's correlation test, the sample is biased. However, 98% of the matches were found in the inner loop of the "region grower" where the correlation threshold is quite low:

THRESHOLD = MAX(.51,AUTOCORR).

This explains the absence of matches whose correlation is less than the auto-correlation, but the trend in each row indicates that not many good matches would be found there anyway. The number of matches found for a given auto-correlation seems to peak where the correlation = (1+AUTOCORR)/2.

Auto-correlation \ Correlation	1.	.95	.90	.85	.80	.75	.70	.65	.60	.55	.50
≥.95	0	0	0	0	0	0	0	0	0	0	0
.90	55	9	0	0	0	0	0	0	0	0	0
.85	43	64	20	1	0	0	0	0	0	0	0
.80	40	73	33	20	0	0	0	0	0	0	0
.75	16	67	48	32	15	1	0	0	0	0	0
.70	5	39	59	44	30	12	1	0	0	0	0
.65	6	44	33	54	25	12	6	1	0	0	0
.60	2	15	35	44	41	30	19	15	1	0	0
.55	2	10	23	49	37	22	19	16	9	8	0
.50	1	5	11	30	30	28	18	17	7	8	0
.45	1	4	6	16	15	13	14	28	14	10	0
.40	0	1	1	6	10	11	13	12	7	6	0
.35	1	0	2	3	7	4	4	9	2	5	0
.30	0	2	1	2	0	0	1	1	2	3	0
.25	0	0	0	1	0	1	2	2	1	1	0
.20	0	1	0	2	1	1	1	0	1	0	0
.15	0	1	0	1	0	1	0	0	0	0	0
.10	0	0	0	0	0	2	0	0	0	2	0
.05	0	0	0	0	0	0	0	0	0	0	0
<.05	0	0	0	0	1	0	0	1	0	1	0

XI. References.

[Baumgart] Bruce Guenther Baumgart, "Geometric Modeling for Computer Vision", AIM-249, Oct 74.

[Grape] Gunnar R. Grape, "Model Based (Intermediate-level) Computer Vision", AIM-201, May 73 (PhD Thesis).

[Hannah] Marsha Jo Hannah, "Computer Matching of Areas in Stereo Images", AIM-239, July 74 (PhD Thesis).

[Horn] Berthold Klaus Paul Horn, "Shape from Shading: a Method for Finding the Shape of a Smooth Opaque Object from One View", PhD Thesis, MIT, June 70.

[Levine] M.D. Levine, D.A. O'Handley, and G.M. Yagi, "Computer Determination of Depth Maps", Computer Graphics and Image Processing (1973),2, pp. 131-151.

[Nevatia] Ramakant Nevatia, "Structured Descriptions of Complex Curved Objects for Recognition and Visual Memory", AIM-250, Oct 74 (PhD Thesis).