Best Available
Copy
for all Pictures

AD 786-720

COMPUTER MATCHING OF AREAS IN STEREO IMAGES

Stanford University Computer Science Department
Stanford, CA

Jul 74

U.S. Department of Commerce
National Technical Information Service
NTIS

STANFORD ARTIFICIAL INTELLIGENCE LABORATORY
MEMO AIM-239

STAN-CS-74-438

COMPUTER MATCHING OF AREAS IN STEREO IMAGES

BY

MARSHA JO HANNAH

SUPPORTED BY

NATIONAL SCIENCE FOUNDATION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
JET PROPULSION LABORATORY
AND
ADVANCED RESEARCH PROJECTS AGENCY
ARPA ORDER NO. 2494

JULY 1974

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

102

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER STAN-CS-74-438	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) COMPUTER MATCHING OF AREAS IN STEREO IMAGES.		5. TYPE OF REPORT & PERIOD COVERED technical, July 1974
		6. PERFORMING ORG. REPORT NUMBER STAN-CS-74-438
7. AUTHOR(s) Marsha Jo Hannah		8. CONTRACT OR GRANT NUMBER(s) DAHC-15-73-C-0435 NGR-05-020-508
9. PERFORMING ORGANIZATION NAME AND ADDRESS Stanford University Computer Science Dept.		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ARPA 2494
11. CONTROLLING OFFICE NAME AND ADDRESS ARPA/IPT, Attn: Stephen D. Crocker 1400 Wilson Blvd., Arlington, Va. 22209		12. REPORT DATE July 1974
		13. NUMBER OF PAGES 100
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) ONR Representative, Philip Surra Durand Aeronautics Bldg., Rm. 165 Stanford University Stanford, California 94305		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

PRICES SUBJECT TO CHANGE

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This dissertation describes techniques for efficiently matching corresponding areas of a stereo pair of images. Measures of match which are suitable for this purpose are discussed, as are methods for pruning the search for a match. The mathematics necessary to convert a set of matchings into a workable camera model are given, along with calculations which use this model and a pair of image points to locate the corresponding scene point. Methods are included to detect some types of unmatchable target areas in the original data and for detecting when (continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

100

a supposed match is invalid. Region growing techniques are discussed for extend matching areas into regions of constant parallax and for delimiting uniform regions in an image. Also, two algorithms are presented to show some of the ways in which these techniques can be combined to perform useful tasks in the processing of stereo images.

STANFORD ARTIFICIAL INTELLIGENCE PROJECT
MEMO AIM-239

JULY 1974

COMPUTER SCIENCE DEPARTMENT REPORT
STAN-CS-74-438

COMPUTER MATCHING OF AREAS IN STEREO IMAGES

by

Marsha Jo Hannah

ABSTRACT: This dissertation describes techniques for efficiently matching corresponding areas of a stereo pair of images. Measures of match which are suitable for this purpose are discussed, as are methods for pruning the search for a match. The mathematics necessary to convert a set of matchings into a workable camera model are given, along with calculations which use this model and a pair of image points to locate the corresponding scene point. Methods are included to detect some types of unmatchable target areas in the original data and for detecting when a supposed match is invalid. Region growing techniques are discussed for extend matching areas into regions of constant parallax and for delimiting uniform regions in an image. Also, two algorithms are presented to show some of the ways in which these techniques can be combined to perform useful tasks in the processing of stereo images.

This research was supported in part by the Advanced Research Projects Agency of the Office of Defense under Contract No. DAHC-15-73-C-0435, in part by a National Science Foundation Graduate Fellowship, in part by the National Aeronautics and Space Administration under Grant No. NGR-05-020-508 Suppliment No. 1, and in part by Jet Propulsion Laboratory sub-contract No. 952489.

ACKNOWLEDGEMENTS

There were certain people and institutions without whom this research would not have been possible. Most important, perhaps, was the Stanford Artificial Intelligence Project, which provided the facilities and atmosphere to encourage research.

Individual thanks go to my adviser, John McCarthy, the creator and director of Stanford A. I. He, Jerry Feldman, and especially Tom Binford put in many hours reading and criticizing the multiple versions of this dissertation.

I am most indebted to Lynn Quam, who laid the programming groundwork for image processing at A. I. and built the video synthesizer hardware, both of which greatly simplified my work. He also acted as a sounding board on whom ideas could be tried out and lent a great deal of moral support. Special thanks to him for repeated proof-readings of this dissertation and for time spent converting my video synthesizer output to the final photographs.

Other people to whom I owe specific thanks include Andy Moorer, who also helped with the final photographs and was of much aid in getting my head together about FFT's. Bob Tucker gave willingly of his time in early attempts to get my video synthesizer output converted to photographs. Bo Eross wrote User Disk Pack code to simplify my handling of massive image data and helped in the optimization of critical code.

Finally, I would like to thank collectively all of the hardware and software hackers at A. I. who created and maintained the facilities. Particular thanks to those responsible for the TV and ET text editors, the SAIL language and compiler, the RAID debugger, the Data Disk television terminals, the Xerox Graphics Printer and its fonts, and A. I.'s PDP-10 time-sharing system.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

Early computer vision research was mainly concerned with operations on pictures--such as encoding, enhancement, and edge detection [Rosenfeld, 1969a]--and with analysis of single images--for example, interpreting images containing bodies from a known set of objects [Roberts, 1963; Guzman, 1968].

Early matching work fell into the domain of pattern recognition--matching a description of an idealized object against descriptions generated from analysis of an image containing that object. Some pixel-by-pixel matching was done in matching a template of an alphanumeric character against pictures of hand-printed characters. (Rosenfeld [1969b and 1973] provides excellent surveys of the literature for single image processing.)

Stereo vision was for a long time the domain of psychologists and physiologists, whose interests were in understanding human stereo vision [Julesz, 1961]. The major use of stereo was in photogrammetry--converting aerial photographs to contour maps, usually by optical methods [Bouchard and Moffitt, 1965].

Eventually, computer stereo image processing became attractive. Julesz [1963] saw it as a method of studying human stereo perception. Computer photogrammetry techniques were developed and used to deal with telemetered image data from satellites. Image processors began to use stereo to determine depth information [Quam et. al., 1972].

All of these applications required efficient ways of matching areas of one picture with the corresponding areas of another, similar picture. Quam [1971] developed a spiraling, stepping algorithm to facilitate his aligning of Mariner spacecraft images for variable feature detection. Barnea and Silverman [1972] reported a sequential decision algorithm which they used in matching weather satellite photos. Other general investigations of matching have been done by Fischler [1971] and by Fischler and Elschlager [1971].

STATEMENT OF THE PROBLEM

What is matching? By matching, we mean the process of finding, for a given sub-area (window) of an image X, the sub-area of image Y which contains point for point the same intensity information. Matching should not be confused with mapping. Mapping implies that there is some general function $(Iy,Jy) = f(Ix,Jx)$ which gives the position of corresponding points in image Y for a given set of points in image X. Matching is a special case of mapping--the case in which the mapping function is a simple translation of axes, $(Iy,Jy) = (Ix,Jx) + (Ti,Tj)$ within the area being matched.

This thesis is concerned with matching, not mapping. Therefore, we are limited to those areas of pairs of images which do not have large perspective changes from one view to the other. This condition is met by small angle stereo and by distant objects in larger angle stereo pairs. We must also exclude areas of images representing objects which themselves move or are moved so as to present differing projections to the cameras. Similarly, we will also limit ourselves to high-quality pictures, i.e., those without scratches or other blemishes on the negatives, those having low noise, etc. These limitations assure that our target areas will have matching candidate areas.

The subject of this thesis is as follows: given two images of a scene, constrained as above, use the information in the pictures to match target area A of image X with its corresponding candidate area in image Y. We will discuss general techniques for matching, efficient methods by which matching can be done, some of the problems that can occur when matching real data, and ways of extending matching areas. In addition, we will describe some of the algorithms which have been implemented to use these techniques.

DESIGN OF THE INVESTIGATION

Picture processing is, for the most part, an applied science. It seeks to show that something is possible, not by formally proving that it can be done, but by doing it. In keeping with this spirit, this dissertation will contain no formal proofs of existence, termination, correctness, or running time. It will contain discussions of techniques and algorithms and reports on how well these techniques work when implemented.

There is, underlying all techniques presented in this thesis, a very basic philosophy. Machine vision will in the near future be used for those tasks which man can do but doesn't want to, such as assembly line drudgery, or those tasks which he wants to do but can't, such as exploring inhospitable planets. In the first case, the structure of the task environment is well known and can be used to make the performance of the task more efficient. This is the problem addressed and approach used by the Hand-Eye group at the Stanford Artificial Intelligence Project [Feldman, 1969]. In the second case, the structure of the environment is only crudely known, hence can only loosely be used to expedite the task.

It is this latter variety of problem for which the techniques of this thesis were to be designed. Consequently, we will avoid whenever possible overspecialization through the use of particular assumed structure or semantics in the completion of our tasks. Our techniques may not be as powerful as those using such information, but they will be more general.

Most of the techniques described in this thesis have been programmed; those which have not will be so noted. The photographic illustrations in this thesis are derived from visual output generated by these programs on a television monitor. No photographic trickery has been done; what the reader sees is roughly what a person operating that program would see on his monitor.

DEFINITIONS

Some of the terms from the field of computer vision which are used in this thesis are defined below.

Picture--a two-dimensional array of integer values which represent the light intensities of a scene at some set of grid points.

Point--one of the array elements of a picture.

Pixel--(contraction of picture element) a point in a picture.

Color picture--a set of three pictures, representing the red, green, and blue filter components of a color photograph or a color television picture.

Image--the set of pictures representing a photograph--one picture for a black-and-white photograph or three pictures for a color photograph.

CONVENTIONS OF PICTURE PROCESSING

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.

The intensities at each pixel are represented by numbers from 0 through $k = 2^n - 1$, with 0 representing no light, or black, and k representing full light, or white. Pixels are stored packed, as many as will fit per word of computer memory.

NOTATIONAL CONVENTIONS

As in normal programming usage, the following compromises with standard mathematical notation have been made.

Square root signs are replaced by the function SQRT.

The raised dot for multiplication is replaced by *.

The following mathematical conventions are used.

Summation signs are indicated by a sigma. The variable which is being summed over is written below the sigma. When exact ranges for the summation are to be given, they are given as a boolean expression in the place of the summation variable. The function being summed is written to the right of the sigma. Parentheses are used only when necessary to avoid confusion.

Examples: $\sum_{i} X_i$ and $\sum_{a \le i < b} X_i$

The mean of a variable is indicated by overbar notation.

$$\overline{X} = \frac{\sum_{i} X_i}{\sum_{i} 1}$$

OTHER CONVENTIONS

Illustrations are numbered with Arabic numerals within chapters and are prefaced by the chapter number, e.g. the first illustration in Chapter 3 is Illustration 3-1. All illustrations for a given chapter appear together at the end of the chapter. Prints of the original data appear in Appendix A.

Equations are numbered with lower case Roman letters within chapters and are prefaced by the chapter number, e.g. the first equation in Chapter 2 is Equation 2-a.

Chapter 2

BASIC AREA MATCHING TOOLS AND TECHNIQUES

Suppose one has been given two digitized photographs which were taken of the same scene, but which differ in some respect, such as point of view. Consider the problem of using a computer to determine which area of picture Y (candidate area) best matches a given area of picture X (target area).

Geometrically, two areas match if they both are projections of the same three-dimensional piece of scene. Intuitively, two areas match if they "look the same". Computationally, two areas match if a calculated measure of match between them is sufficiently optimal.

CORRELATION

Since we are dealing with the probability of a match occuring, some statistical measure is desirable as the measure of match. The common measure for this is discrete correlation,

$$COR = \sum_i X_i * Y_i$$

which can be normalized by the means of the samples

$$COR = \sum_i (X_i - \bar{X}) * (Y_i - \bar{Y})$$

or by the second moments of the samples

$$COR = \frac{\sum_i X_i * Y_i}{SQRT(\sum_i X_i^2 * \sum_i Y_i^2)}$$

or by both.

$$COR = \frac{\sum_i (X_i - \bar{X}) * (Y_i - \bar{Y})}{SQRT(\sum_i (X_i - \bar{X})^2 * \sum_i (Y_i - \bar{Y})^2)} .$$

The last is the nicest to work with, since is has an absolute value less than or equal to one, and its absolute value equals one if and only if $X_i = a*Y_i + b$ for all i.

DIFFERENCE MEASURES

Also used are measures based on the difference between the samples over the two areas, such as root-mean-square error,

$$RMS = SQRT\left(1/n \sum_i (X_i - Y_i)^2 \right)$$

which can also be normalized by the means of the samples.

$$RMS = SQRT\left(1/n \sum_i ((X_i - \bar{X}) - (Y_i - \bar{Y}))^2 \right) \qquad (2\text{-a})$$

Absolute difference is also used.

$$AD = \sum_i |X_i - Y_i| /n$$

It too can be normalized by the means.

$$AD = \sum_i | (X_i - \bar{X}) - (Y_i - \bar{Y}) | /n$$

The calculation of normalized absolute difference, however, requires two passes over the data--one to calculate the sample means and one to sum the absolute differences which include these sample means. All other measures mentioned here, including normalized correlation and normalized RMS, can be calculated in one pass over the data. What distinguishes normalized absolute difference from the rest is the presence of summations both inside and outside of the absolute value sign. Absolute value is not a linear operator, therefore effectively foils the algebraic manipulations of summations which permit the other normalized measures to be calculated in one pass. Because of the added inconvenience of a second pass over the data, normalized absolute difference is rarely used.

Both RMS and absolute difference yield values between zero and a number bounded by the largest difference between the samples, which is in turn bounded by the maximum possible intensity at a pixel.

COMPARISON OF THE MEASURES OF MATCH

Perhaps at this point a few words should be said about the relative merits of correlation, RMS, and absolute difference as measures of match.

RMS and absolute difference are clearly related. There is also a relationship between normalized RMS and normalized correlation. In the following, let

$$T(X,Y) = \sum_i (X_i - \bar{X}) * (Y_i - \bar{Y}) \quad .$$

Correlation can now be expressed as

$$\mathrm{COR} = \frac{T(X,Y)}{\mathrm{SQRT}(\ T(X,X) * T(Y,Y)\ )} .$$

Equation 2-a expands to

$$\begin{aligned} \mathrm{RMS} &= \mathrm{SQRT}\Big(\ 1/n \sum_i \big(\ (X_i - \bar{X})^2 - 2*(X_i - \bar{X})*(Y_i - \bar{Y}) + (Y_i - \bar{Y})^2\ \big)\ \Big) \\ &= \mathrm{SQRT}\big(\ 1/n\ (\ T(X,X) - 2*T(X,Y) + T(Y,Y)\ )\ \big) . \end{aligned}$$

Hence, we have

$$\mathrm{COR} = \frac{T(X,X) + T(Y,Y) - n * \mathrm{RMS}^2}{2 * \mathrm{SQRT}(\ T(X,X) * T(Y,Y)\ )} .$$

Being related, correlation, RMS, and absolute difference might be expected to give similar results when used as the measure match.

The cheapest measure of match, in terms of the number of instructions required to implement it, is absolute difference. Two samples which match exactly have an absolute difference of zero. It may be the case, however, that the pixel intensities in the candidate area equal those in the target area plus a constant (offset), that is, $Y_i = X_i + b$. In this case, the absolute difference between the two intuitive matching areas would be non-zero, perhaps greater than the absolute difference for some other area which is similar, but not intuitively the matching area.

Normalized RMS takes care of this problem by subtracting the means of the two areas from each of the intensity values within the samples. It trades a little more time in the calculation of the measure of match for more flexibility in its application.

Suppose, however, that the pixel intensities from the matching area are equal to a constant factor (gain) times the intensities from the target area, plus some constant offset, that is, $Y_i = a*X_i + b$. The value of RMS over matching areas in this case is non-zero. This can result in rejection of a matching area should some non-matching but relatively similar area contain data which has a relative gain of one.

Normalized correlation, although more expensive, is designed to handle both a constant gain and a constant offset. Subtracting the means removes the problem of the offset; dividing by the variances takes care of the gain. This can lead to multiple match candidates if several areas of different relative gains and offsets resemble each other. However, this merely introduces impostors, it does not discard true matches.

Because relative gain and offset are frequently present in digital stereo images, the author prefers normalized cross-correlation to the other measures of match, and has developed matching techniques centered around correlation. However, if gain and offset are not a problem, or are known and can be taken into account in the calculation of the difference measures, then the techniques presented in this thesis can be adapted to normalized RMS or absolute difference. Since the techniques of this thesis were developed and originally implemented with correlation, they are discussed in terms of correlation.

FAST FOURIER TRANSFORMS FOR CONVOLUTION

Fast Fourier convolution is often mentioned as a tool for matching. It is a method for calculating the Σ XY term used in correlation and RMS error somewhat more efficiently.

This Σ XY term is the discrete convolution of the two samples; the Fourier transform of this convolution is equivalent to the product of the Fourier transforms of the two samples. Thus it is possible to do the summation by taking the transforms of the two samples, multiplying them, then taking the inverse Fourier transform of the result. If this is done for a target area out of picture X and all of picture Y, the result is an array, each element of which contains the value of the convolution between the candidate area centered at that point and the target area.

With the fast Fourier transform, it is possible to do a transform of a sample of $m = 2^n$ points in time proportional to m log2 m [Singleton, 1967]. Let N be the maximum dimension of picture X and W be that of the window being matched. Due to the aliasing problem, it is necessary that m be not less than N+W [Cooley, et. al., 1967], as well as being a power of two. If we let L be the constant necessary to bring N+W up to 2^n, then the Σ XY for N^2 correlations can be done in time proportional to $(N+W+L)^2 \log2 (N+W+L)^2$ by the FFT method, as compared to time proportional to N^2W^2 for the direct computation. Of course, for normalized correlation or normalized RMS, it is still necessary to compute Σ X, ΣX^2, Σ Y, and ΣY^2 directly, and to combine them in order to calculate each of the measures of match. Employing sliding sums to calculate these terms adds time proportional to N^2; time proportional to N^2 is also added to combine the sums for calculating the measures of match.

Which method is faster for a given problem will depend on the valuee of N and W and the constants of proportionality, which depend on the implementations. Illustration 2-1 compares the FFT approach with the direct approach for the implementations used at Stanford A.I. and several values of N and W.

AREA SAMPLING

Considerable time is wasted in calculating the measure of match over all the pixels in every candidate area in the second picture. Like most searches, the search for a match spends most of its time failing--calculating the measure of match for areas that don't match. If one can reduce the amount of time spent failing, a significant saving will result.

Barnea and Silverman [1972] observed that, for most candidate areas, it becomes obvious after a small fraction of the points in the area have been processed that the measure of match is going to have a non-optimal value. If processing of that area is aborted when the area's non-optimality is discovered, a considerable savings of time results.

Toward this end, they propose the following sequential decision algorithm. Start calculating the measure of match, taking corresponding pairs of sample elements out of the two areas in pseudo-random order. At intervals, monitor the value of the measure of match. If at any time the measure is non-optimal according to their decision criteria, discontinue the calculations and discard the area as non-matching. Otherwise, continue adding in samples randomly until either the whole area has been included or the measure becomes non-optimal.

Barnea and Silverman claim that this algorithm is up to 50 times faster than matching by ordinary correlation techniques. Unfortunately, they do not separate the savings due to their using absolute difference as the measure of match from the savings due to the algorithm itself. Quam [unpublished research, 1973] finds, in one particular application, that their algorithm used with normalized RMS is five to ten times as fast as ordinary normalized correlation techniques.

Reducing the number of points handled in some of the sample areas is one side of the coin. The other side represents the possibility of not calculating that measure of match for every candidate area in the second picture.

Direct method. Tabulated values are 0.000026 N^2 W^2 seconds.

N \ W	11	13	15	17	19	21
100	31.4600	43.9400	58.5000	75.1400	93.8600	114.6600
150	70.7850	98.8650	131.6250	169.0650	211.1850	257.9850
200	125.8400	175.7600	234.0000	300.5600	375.4400	458.6400
250	196.6250	274.6250	365.6250	469.6250	586.6250	716.6250
300	283.1400	395.4600	526.5000	676.2600	844.7400	1031.9400
350	385.3850	538.2650	716.6250	920.4650	1149.7850	1404.5850
400	503.3600	703.0400	936.0000	1202.2400	1501.7600	1834.5600
450	637.0650	889.7850	1184.6250	1521.5850	1900.6650	2321.8650
500	786.5000	1098.5000	1462.5000	1878.5000	2346.5000	2866.5000

FFT method. Tabulated values are 0.000080 * 4(N+W+L)^2 log2(N+W+L) seconds, ie. 2 FFT's--one for the window, one to inverse FFT the product of the FFT of the window and the FFT of Picture Y. This neglects the time needed for N^2 complex multiplies to form the product.

N \ W	11	13	15	17	19	21
100	36.7002	36.7002	36.7002	36.7002	36.7002	36.7002
150	167.7722	167.7722	167.7722	167.7722	167.7722	167.7722
200	167.7722	167.7722	167.7722	167.7722	167.7722	167.7722
250	754.9747	754.9747	754.9747	754.9747	754.9747	754.9747
300	754.9747	754.9747	754.9747	754.9747	754.9747	754.9747
350	754.9747	754.9747	754.9747	754.9747	754.9747	754.9747
400	754.9747	754.9747	754.9747	754.9747	754.9747	754.9747
450	754.9747	754.9747	754.9747	754.9747	754.9747	754.9747
500	754.9747	3355.4432	3355.4432	3355.4432	3355.4432	3355.4432

Illustration 2-1. These tables compare the relative efficiencies of the direct method and FFT for calculating the convolution Σ XY of a window of W^2 points with a picture of N^2 points. The constants of proportionality are derived from machine language codings on the PDP-10 at Stanford A.I. by Lynn Quam (direct method) and Don Oestereicher (FFT).

Chapter 3

SEARCH STRATEGIES AND REFINEMENTS

The idea of shortening a search by "pruning" the search space is not a new one. Heuristic search has been a part of artificial intelligence from the beginning [Nilsson, 1972]. The basic idea is simple: arrange the search in such a way that entire sets of solutions are considered at once. Attach to each set some way of measuring whether or not it has a good chance of containing the desired solution. Work in detail on only those sets which show promise. Whenever possible, work first on those sets which show most promise.

With most pictorial data, there is a fair amount of local coherence. By this, we mean that an area centered at one pixel does not usually differ greatly from an area centered at a neighboring pixel. An alternate expression of this would be to say that most pictorial data consists primarily of low frequency information. This makes it possible to use one candidate area as a representative of a set of areas centered at adjacent points. The evaluation of some computationally inexpensive measure of agreement over the representative area serves as the evaluation of the set. A number of variations on this technique can be used in pruning the search for a match.

GRIDDING

Consider for a moment the surface formed by plotting correlation as a function of position of candidate area centers in the vicinity of the matching candidate area. Because of the local coherence of most target areas, this correlation surface usually falls off gradually as one moves away from the matching area's center. (See Illustration 3-1). Therefore, in the immediate vicinity of the peak in a correlation surface which indicates a match, the correlations will usually be above some threshold.

One can take advantage of this fact by calculating the correlations between the target area and candidate areas centered at points on an n by n grid over picture Y. For suitable n and threshold, it is clear that one of the grid candidates must lie somewhere on the match peak above the significance threshold. By searching in detail the immediate vicinity of any grid candidate showing a significant correlation, one can locate the match peak. This is the technique of gridding.

If one uses an n by n grid on an N by N picture and finds k correlations above the significance threshold Q, one calculates about $(N/n)^2 + k*n^2$ correlations of area W^2 in finding the match. In comparison, the direct method requires N^2 correlations to locate the match. Since in most cases, k is small, gridding results in a savings of a factor of n^2 over the direct method.

The success of gridding, of course, lies in the choice of n and of Q, which influences k. Examining the first pair of correlation cross sections in Illustration 3-1, we see that for Q=.5, n must be 1, but if Q=.1, n can be 5. For the second pair, Q=.5 means n=6, and Q=.1 means n=10. The allowable values for n and Q are not only interconnected, but also depend on the individual correlation peak.

However, when we begin our search for a match and are ready to set n and Q, we do not yet know what the correlation peak will look like. We do know that, under ideal conditions for matching, the target area will very closely resemble its match. If the matching area exactly duplicated the target area, then the correlation surface would be identical to the autocorrelation surface. (See Appendix B.) In practice, this precise equivalence does not hold; however, the correlation and autocorrelation surfaces do resemble each other (See Illustration 3-2). Hence the autocorrelation peak can give a good indication of the proper n and Q for a given target area. Extracting this information can be done by inspection, by fitting a second order surface to the correlation peak and measuring its parameters, or by examining the Fourier transform of the data.

In theory, gridding will always work, since the worst it can do is degenerate to the standard method of evaluating the correlation at every point when n=1. In practice, however, gridding is not used if the autocorrelation peak indicates a grid spacing of 1 or 2. Such an autocorrelation peak can occur if the target area contains mainly high frequency information, as is the case in the distant treee along the skyline in the Lab pictures. (See the copies of the original data in Appendix A, coordinates (112,20)in the first image and (113,26) in the second, window radius=7.) It also occurs in extremely noisy images and is a feature of some artificially generated images [Julesz, 1961 and 1963].

REDUCTION

One technique for utilizing local coherence to make the amount data to be handled more manageable is reduction [see e.g. Kelly, 1970]. In our application, this means making a new pair of picturee by spatially reducing the originals--effectively replacing m by m squares of pixels by one pixel having the average intensity of that square. Appropriate areas are then matched in the reduced pictures. Finally the correlation peaks for the areae found to match best in the smaller pictures are searched in the original, higher resolution pictures.

Doing an m by m spatial reduction on the pictures means that there are now $(N/m)^2$ potential areas for the reduced target area to match instead of N^2, a savings of a factor of m^2 in the number of correlations to be calculated. If the target area is to represent the same objects, then its size is also reduced from W^2 to $(W/m)^2$ pixels. This results in a savings of a factor of m^2 in the correlation calculation loop, for an overall savings factor of m^4.

If the target area is not very big to start with, reducing the images may cause the target area to no longer represent a valid statistical sample. If one is not constrained to matching any particular area, but can enlarge the effective area to maintain a valid sample size in the reduced pictures, then reduction can be used. The savings factor will depend on the exact size of the window which must be used, but should be somewhere between m^2 and m^4.

As with gridding, there is an additive term of $k*m^2$ full scale correlations necessary to determine the location of the unreduced match. Here, k depends on how many areas within the reduced second image will resemble the reduced target area, which is difficult to predict. There is also the overhead of reducing the two images, but this can often be combined with some other necessary processing.

The success of reduction depends on the choice of m, which in turn depends on the information within the picture. Intuitively, if most of the information in the picture lies in features which are p pixels wide, then one does not wish to reduce the picture by a factor of p or greater. Computationally, if the Fourier transform of the picture reveals that a significant part of the power is in spatial frequencies higher than N/p, one should reduce the picture by a factor of less than p. In general, one should avoid reduction by a factor sufficiently large to change the spatial frequency or information content of the pictures. One way to check on this is to examine the autocorrelation peak in both the original and reduced pictures. If the peak is much narrower in the original than in the reduced image, too much reduction has happened.

If one allows the choice of m to be determined by the data, then in theory, reduction will always work, since it simply degenerates to the standard method for m=1. If a larger than recommended reduction is employed--for example to decrease noise--then the possibility exists that the technique of reduction will fail to produce the proper match.

SIMILARITY

The technique of similarity differs from previously described techniques in that it does not use correlation as the basis for pruning the search for the match. The idea behind similarity is simple--if two areas match, then statistical measures calculated over them, such as means and variances, should be similar.

To employ the basic technique of similarity, one first calculates a vector of statistics for the target area and for each of the candidate areas. The most promising candidate areas are those which have vectors of statistics similar to the vector for the target area, as determined by a weighted distance metric. Then the correlation values between the target area and those candidate areas are used to decide which promising candidate area is the matching area.

Comparing similarity to the standard method is not as simple as comparing gridding or reduction. We can no longer just count the number of correlations calculated, since most of the time involved in using similarity is spent doing things other than correlating.

Calculating the statistics over N^2 areas in picture Y with sliding sums will require time proportional to N^2. The constant of proportionality will, of course, depend upon how many statistics are calculated and upon the statistics themselves. For instance, on the PDP-10, it takes 0.525 ms per pixel to calculate 5 statistics--mean and variance of intensity and vector mean (2 components) and variance of color--for a color image. It takes 0.145 ms per pixel to calculate 2 statistics--mean and variance of intensity--for a black-and-white image.

Comparing the r statistics in the target vector to the r statistics in N^2 candidate vectors will require time proportional to $r*N^2$. Example: it takes 0.175 ms to compute a weighted distance metric for 5 statistics and store the resulting distance; it takes 0.075 ms for 2 statistics. Sorting n distances to order the areas by how promising they are requires $0.070*(\log n)*(n + \log n)$ ms. Finally, calculating the correlations for the k most promising areas, using a window of area W^2, requires $0.065*k*W^2$ ms. By comparison, it takes $0.065*N^2*W^2$ ms to calculate all of the correlations directly.

To better illustrate the comparison, consider matching a 21 by 21 area out of a 150 by 150 picture, let k=10, and use the 5 component vectors from color images. For this example, it would require about 645 seconds to compute the correlations necessary to determine the match directly; similarity spends about 11.8 seconds calculating the vectors, 3.9 seconds calculating the distances, 25.2 seconds sorting them, and 0.3 seconds calculating the k correlations, for a total of about 41.2 seconds, representing a savings factor of about 16.

As savings factors go, 16 is neither trivial nor wonderful. So far, however, we have implemented similarity in a brute force style comparable to the direct method for finding the match. It is possible to refine similarity in order to make it much more efficient.

We have pointed out before that most images have a local coherence which causes area-based measures such as mean and variance to change slowly as the area center is moved by one or two pixels. This means that we really do not need to calculate the distances between the target area vector and vectors for areas centered at every point in the second image. We can allow an area centered at one point to represent those areas centered at adjacent points and apply heuristic search methods.

For instance, one could sort only those vectors of statistics which fall on an m by m grid, reducing the number of distances which must be calculated and sorted to $(N/m)^2$. Then, from the most promising k such grid points, one could hill-climb in the vector distance space until one found the most promising vectors, which would be checked via correlation to determine the match. Here we spend the same amount of time calculating the vectors, but only $0.175*(N/m)^2$ ms calculating distances, $0.070*(\log2 (N/m)^2)*(\log2 (N/m)^2 + (N/m)^2$ ms sorting the distances, $0.175*k*m^2$ ms calculating distances for the hill climb of promising vectors, and $0.065*k*W^2$ ms doing the correlations for these promising hill tops.

Suppose that we set N=150, W=21, k=10 as before and let m=10. As before, we spend 11.8 seconds calculating the vectors, 0.04 seconds calculating grid point distances, 3.13 seconds sorting these distances, 0.18 seconds calculating distances for the hill climbs, and 0.30 seconds doing the correlations for these promising hill tops. This is an overhead of 11.8 seconds, plus 0.65 seconds per match. For only one match, this gives a savings factor of slightly over 50. If the overhead is spread among 20 matches, the savings factor goes up to over 500!

This technique has not been implemented, however, because of the large amount of storage memory it requires. In addition to the 150^2 6-bit intensity values of the second image (which amounts to 3,750 computer words), that are needed for the brute force correlation method, this method also requires $5*150^2$ 36-bit numbers to store the vectors for the second image. This amounts to 112,500 additional words of computer memory, which on most systems is hard to come by. Our speedup of a factor of 500 is accompanied by a very large increase in the space required to do the job.

Now, instead of keeping all of our vectors of statistics from every point, we only keep them for areas centered on an m by m grid over the second picture. The most efficient way to do this for the general case is still with sliding sums. Recording only every m-th vector in both directions means that this now takes $0.065*N^2 + 0.080*(N/m)^2$ ms for the black-and-white and $0.340*N^2 + 0.185*(N/m)^2$ ms for the color vectors described earlier. This time we calculate $(N/m)^2$ distances and sort them as before. For the best k distances, we employ some form of local correlation search to cover that m by m area, which potentially holds the match.

For N=150, W=21, k=10, and m=10 as before, we now spend 7.69 seconds forming the vectors. For each target area, we spend 0.04 seconds calculating the distances and 0.13 seconds sorting them. If we calculate all m^2 correlations for each of the k promising areas, we will spend 28.67 seconds in the correlation loop. Employing gridding or some other form of efficient correlation search can reduce this term significantly. Realistically, if we share the overhead among 20 matches and do about 150 correlations in searching the most promising areas (see Illustration 3-3), then matching one target area will take around 4.85 seconds, a savings of a factor of approximately 130 over the direct method. The extra space required is a mere $5*15^2$ 36-bit words, or 1,125 words, a reasonable amount.

Clearly, similarity is a very complicated technique whose relative efficiency depends on a great number of things. The overhead depends heavily on the number and type of statistics used, which will depend on the data and the ingenuity of the experimenter in using it. An increased number of complex statistics makes the overhead greater and increases the amount of time spent calculating the distance measures. But, as Illustration 3-3 shows, having more statistics in the vector can reduce the number of areas which look promising, hence the number of correlations which must be calculated.

The type of statistics used can affect the success of similarity. Averaging measures such as mean and variance have the advantage of being quick and easy to calculate, fairly insensitive to noise, and, as noted before, usually insensitive to small changes in position. In general, statistics that average are prefered to those that count or those that difference.

The calculation of the distances for the vectors and the sorting of the vectors depend on the number of representative areas, hence on the gridding over which the representative areas are taken. Too small a gridding results in a large number of vectors to be compared and sorted; too large a gridding may let the matching area go unrecognized because it fell between two representative areas which didn't resemble it. As with the grid spacing for correlation gridding, the best way to set this grid spacing is to examine the vector surfaces for the neighborhood of the target area.

The technique of similarity usually works, but not always. If the pictures are very homogeneous, all areas will be similar, resulting in many candidate areas to be searched via correlation, hence little savings. If the pictures have much fine detail or are noisy, then the candidate gridding may be so fine that the technique loses its usefulness.

The presence of objects which have moved relative to their backgrounds in the second image may cause the technique of similarity to fail completely for some target areas. For instance, consider the pair of areas in the barn pictures (see Appendix A for the originals) which are centered in the trees to the left of the telephone pole, and have the pole itself in the right half of the areas. These areas will match very well. However, if it should happen that the representative area which has its center physically closest to that of the matching area contains a part of the foreground post, it will not be similar to the target area. Because that representative area is not similar, it will not be searched and the match will be missed.

Indeed, any condition which causes the matching area to require a finer similarity grid than the target area will endanger the success of similarity.

CAMERA MODELS

So far, we have been discussing methods of reducing the search which do not assume anything not directly contained in the picture data. This was the case for our data; however, in general we will know somewhat more about our pictures. A reasonable design constraint on a picture-taking system is that it record how it was oriented when it took the pictures. This information enables one to model the relative positions and orientations of the cameras.

If complete camera model information is not known, as it was in our case, it still is possible to derive a workable model from the pictures themselves. Several things are known to be undecidable given just the information in the pictures. Absolute position, for instance, is not derivable; it requires external knowledge such as measurements made when the picture was taken or recognition of some landmark in the picture. Likewise, it is impossible to say exactly how large or how far away a given object is without measurements or landmarks to establish scale.

It is possible, however, to derive relative positions and relative sizes for objects in the pictures. This is done by assigning an arbitrary position and orientation to one of the cameras and by fixing some distance, such as the distance between the cameras. With these hypotheses and a suitable number of point-pair matches derived by the previously mentioned techniques, the relative orientations of the cameras and positions of objects which appear in both pictures can be calculated.

Theoretically, if one has N unknowns in the camera model and N constraints in the form of matching point pairs, one can obtain a closed form solution for the camera model. In practice, the constraining equations do not usually permit analytical solution. Therefore, a more common technique is to approximate the unknowns by least-squares techniques, either in closed form or by numerical methods. By either method, one needs at least N/2 point pairs. The locations of these point pairs within the images and the location in 3-space of the points they represent is important. If these point pairs are concentrated in one area of the image or if they represent 3-dimensional points which are all coplanar, then N/2 point pairs is not sufficient. For numerical least-squares approximation of the camera model parameters, the author likes to have at least twice as many point pairs as there are parameters to be derived, and to have these pairs well distributed in both images.

Several different approaches have been taken to the problem of deriving camera models from picture information. (See, e.g. [Sobel, 1970])

Since this author was faced with pictures for which no camera model was given and since no available model derivation code was applicable, yet another camera model derivation method has been developed.

This author's approach is based on searching for the camera model which minimizes a least-squares measure of camera model error. Each pair of matching areas is first characterized as a pair of points--the centers of the areas. For every proposed camera model in the search, each pair of points is placed on the image planes of the cameras, and the rays from the principal points of the cameras through these image plane points are calculated. The error is a function of how close these pairs of rays come to each other in 3-space, normalized by the mean distance to the point of approach. (A mathematical explanation of this measure appears in Appendix C.)

This author, not being a numerical analyst, implemented a very unsophisticated function minimizer to search for the best camera model for a given set of points. That program showed that the technique would work, but was slow and unreliable. The calculations presented in Appendix C have since been re-programmed by another student, Donald Gennery, whose program works very reliably and quite fast. It is his program which has derived most of this author's camera models.

For the purpose of limiting the search space, it matters not whether the camera model is given or derived. The existence of a camera model makes possible another search-reduction technique.

With a camera model, it is possible to constrain the search for the matching area to a line in the second image. To do this, the target is characterized by a point, usually its center of mass. This point is projected through the first camera as a ray in 3-space. The 3-dimensional point corresponding to the original point in the image plane must lie on this ray. The ray is now back-projected into the second camera becoming a line segment on the second image plane (whose exact equation is derived in Appendix C). Since the 3-dimensional point was on the ray, its projection into the second image plane must lie on this line segment.

With this knowledge, it is not necessary to search the entire picture for a match; searching along the line segment will suffice. Illustration 3-4 shows for two different areas of the barn pictures a target area in the first image, its center point, the line which this point projects to, and the matching area found by searching along the line segment. This technique reduces the search space in an N x N picture from the N^2 candidate areas centered at the points of the picture to the N or fewer candidate areas centered on the points of the line segment. Performing a one-dimensional analog of the technique of gridding along the line can result in an additional savings of a factor of m, the grid spacing.

Techniques involving camera models will work whenever a camera model exists, but their efficiency in reducing the search depends on the accuracy of the camera model. An exact camera model will give the line exactly. A moderately inaccurate camera model will usually put the line in the right area, although some local searching may be necessary. The better the model, the smaller the local search.

WORLD MODELS

If, in addition to a camera model, there exists a model for the world, then it is possible to predict precisely where the center point of the matching area will be. The ray from the first camera will intersect the world model at a 3-dimensional point which can be back-projected into the second camera, giving the center point of the predicted match.

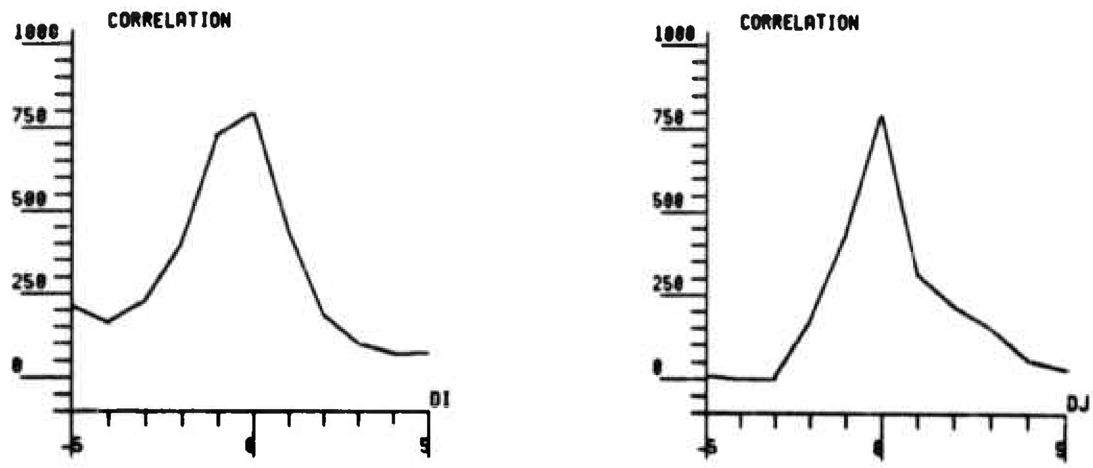
Even a fragmentary world model can reduce the search significantly. For instance, knowledge of the position of the ground plane limits the depth at which an object can lie [Falk, 1969]. Thus the matching center point is constrained to lie on that part of the back-projected line segment between the points which represent the camera and the ground plane.

If the world model is not given, it is still possible to derive it from the matched area pairs. However, derived world models are more often the result of the matching process, not the means for its improvement.

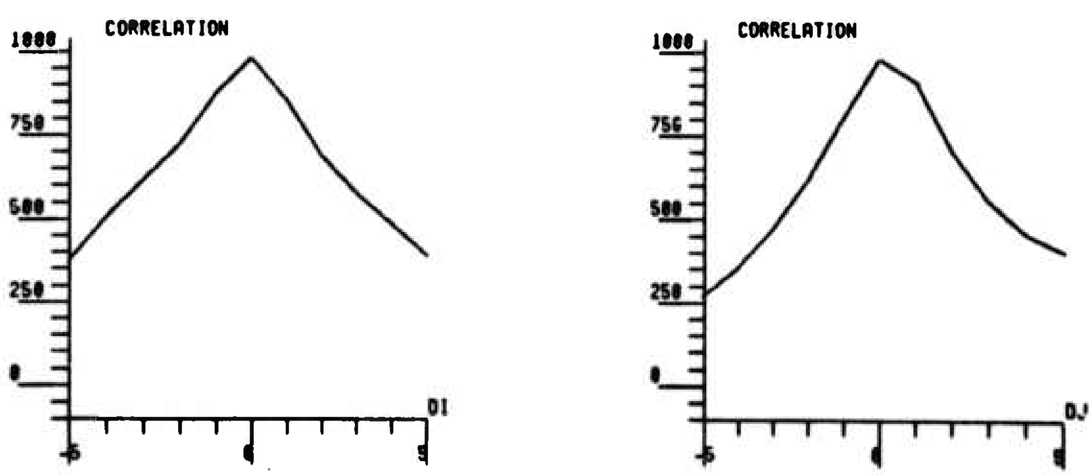
One trivial sort of world model which does not require a camera model (although it can be used with one) is the continuity assumption. It consists merely of assuming that if areas A and B are adjacent in the first image, then their matches will be adjacent in the second image. This, of course, reduces the search space considerably--to the immediate neighborhood of the last match found.

How effective the use of world models is depends on the accuracy of the model. Small errors in the model may make little difference in the predicted position of the match. Large errors, like assuming continuity near a depth discontinuity will cause no match to be found. In this case, a retreat must be made to one of the more general techniques of matching.

Each of the techniques described in this chapter results in a fairly large savings when matching areas in stereo images. Combining two or more of them increases the savings. The author has had excellent success with programs combining reduction, similarity, and gridding and with ones combining reduction, camera models, and gridding. (See Chapter 6 for descriptions of some of the programs.) The author has not implemented any of the world models save the continuity assumption (see Chapter 5), but the Hand-Eye group at Stanford A.I. uses the ground plane model to good advantage, and Bruce Baumgart [unpublished research, 1972] has done some work with exact world models.



LAB PICTURES, (25, 85) - (38, 91)



BARN PICTURES, (85, 65) - (100, 58)

Illustration 3-1. Two sets of correlation cross sections, graphing C(X,Ix,Jx;Y,Iy+dI,Jy) against dI and C(X,Ix,Jx;Y,Iy,Jy+dJ) against dJ. (See Appendix B for an explanation of the notation.) Most correlation surfaces are like these, falling off gradually as one moves away from the match peak.

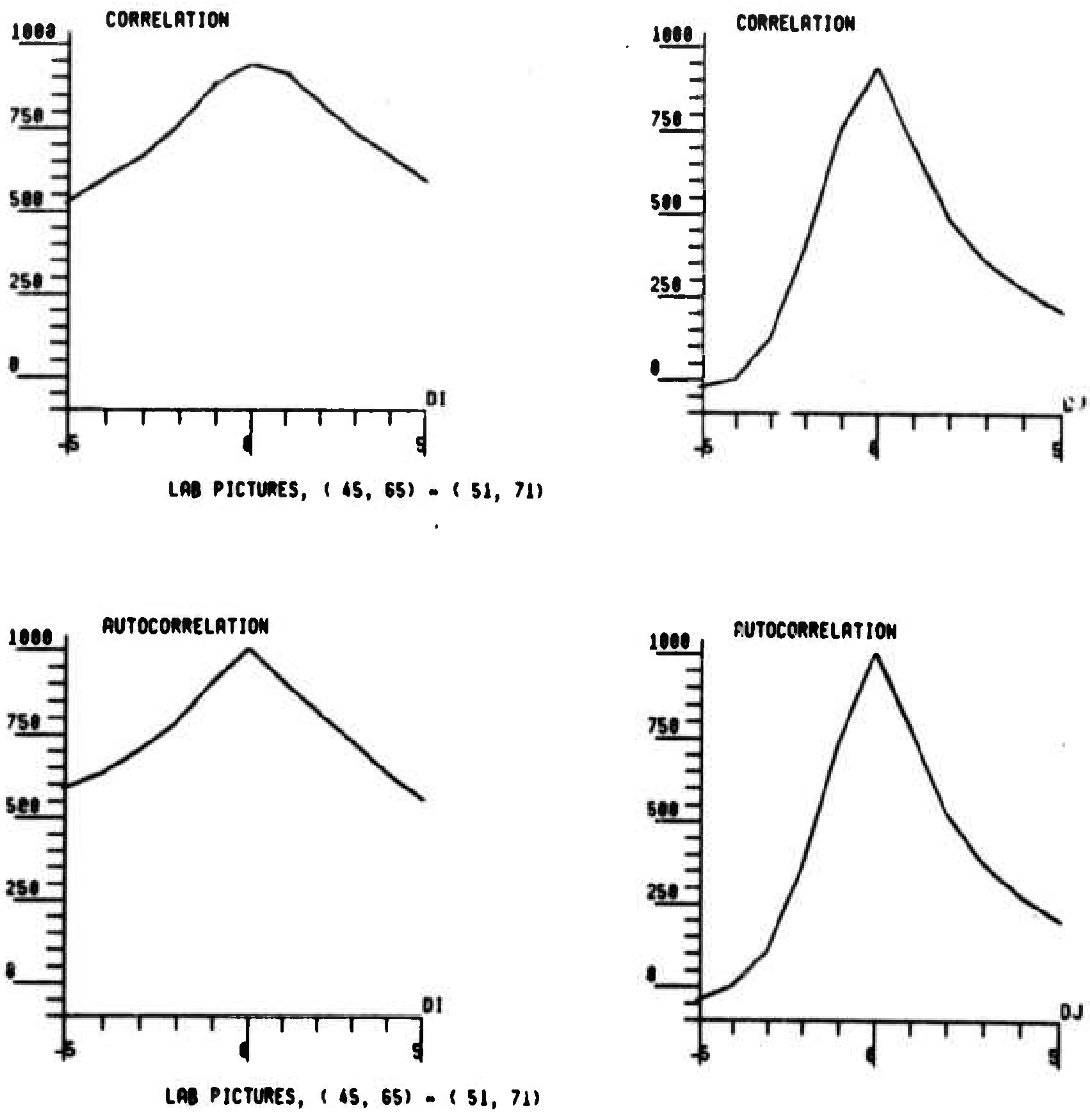


Illustration 3-2. The top row contains graphs of $C(X,Ix,Jx;Y,Iy+dI,Jy)$ against dI and $C(X,Ix,Jx;Y,Iy,Jy+dJ)$ against dJ, as before. Bottom row contains graphs of $C(X,Ix,Jx;X,Ix+dI,Jx)$ against dI and $C(X,Ix,Jx;X,Ix,Jx+dJ)$ against dJ, i.e. the autocorrelation cross sections for the same target area. Like these, most match correlation closely rssemble their autocorrelation peaks.

LAB PICTURES AREA		INTENSITY		COLOR	
IX	JX	AREAS TRIED	COR. CALC.	AREAS TRIED	COR. CALC.
145	25	1	30	1	30
85	25	11	523	3	124
65	25	2	85	1	42
25	25	10	675	4	252
65	45	5	242	3	140
25	5	6	338	2	92
105	5	9	408	3	126
45	5	1	61	1	61
25	85	2	114	2	135
65	5	4	182	3	188

Illustration 3-3. Tabulated results of correlation searches using the search reduction technique of similarity with correlation gridding in the promising areas. The first two columns give the area center in the first Lab picture; the second two tell how many promising areas were found and how many correlations were calculated for the black-and-white vectors described in the text; the third two give the number of promising areas and correlations calculated for the color vectors described. The color vectors are usually worth the increased time needed to calculate them. Either set of vectors results in a significant reduction in the amount of time needed to find the match.

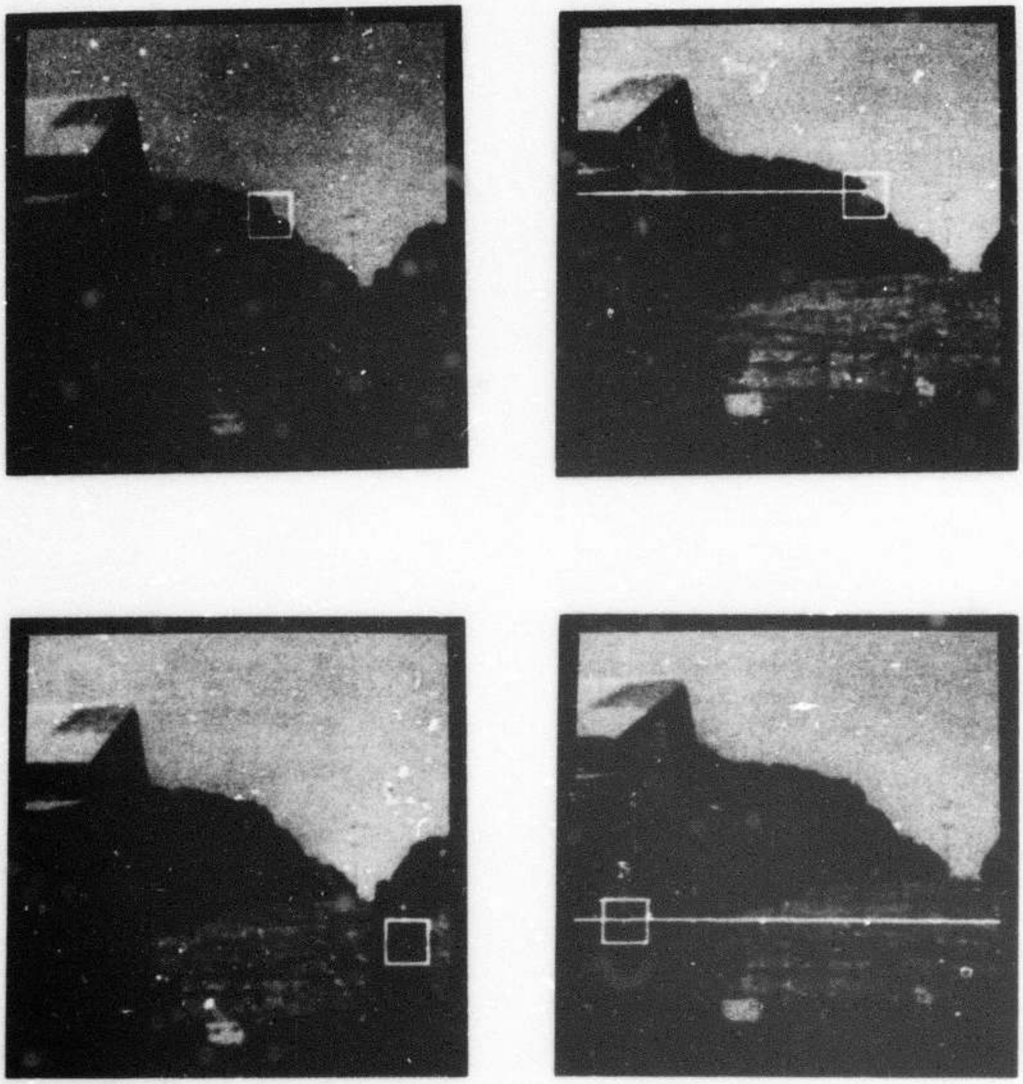


Illustration 3-4. Two pairs of barn pictures, showing camera model searches for a match. In each pair, the first image shows the target areas with their central points; in the second image the line to which the center point projects is shown, along with the matching area and its center.

Chapter 4

UNMATCHABLE TARGET AREAS

Careful analysis of the techniques discussed so far will show that, in addition to the assumptions stated in Chapter 1, we have been making one other, unstated assumption. We have assumed that there exists some window-based algorithm by which all target areas can be matched.

Unfortunately, there are entire classes of target areas which do not fit this assumption, i.e., which require global techniques to determine which area is their intuitive match. These fall into two major groups--those which can be detected before matching is attempted and those which come to light only when matching fails.

DETECTABLE BAD TARGETS

The first group of unmatchable target areas are those containing data which is by its very nature unmatchable. These unmatchable areas can be detected before matching is attempted by examining the target data.

Low Information

When the target area contains little or no information, matching that area is impossible by area-based measure-of-match techniques. For example, consider a window taken out of the cloudless sky of the barn pictures. Based on just the information in that window, it is impossible to say precisely which piece of sky in the second image matches this area. In the absence of noise, a low-information target area will match almost any low-information candidate area, for there is nothing in the target to distinguish which candidate it really matches.

An area of low information is an area of low variance. This is perhaps the most computationally expedient way of determining whether or not an area has sufficient information to be matchable. In the presence of noise, this technique may fail, since noise creates variance. In this case, some other test, such as the presence of an edge, should be used.

An area of low information will also have an autocorrelation peak which, except for a value of 1.0 at zero displacement, will be almost flat. (Illustration 4-1 shows the correlation and autocorrelation graphs for an area of the sky in the barn pictures.) This flatness can be recognized by inspection of the peak, or, if more precise determination is desired, a bivariate normal distribution surface [Freund, 1962] can be fitted to the peak and the parameters of the curve examined. Any area having a very flat autocorrelation peak is unsuitable for matching.

Linear Edge

When the target area has a single linear edge across it with little or no information on either side of this edge, matching is very difficult. An attempt to match such a target will show that the area matches quite well with candidate areas all along the edge in the second image.

This condition is observable in the autocorrelation peak. (See Illustration 4-2) If one fits a bivariate normal distribution surface to the autocorrelation and examines the contours of this surface, one discovers that the peak is really a ridge aligned with the edge.

If we use only the information in the target area, there is nothing to resolve which candidate along the edge is the real match. Target areas displaying this property must be regarded as unmatchable unless further information, such as a camera model or a set of other matches to tie to, is available.

Pre-processing of a target area to determine whether or not it is suitable for matching is expensive. However, if one compares this expense with the expense of searching futilely for a match, such pre-processing becomes worthwhile.

TARGETS WHICH DO NOT MATCH

The second group of unmatchable target areas are those whose counterparts simply do not exist in the second image, due to relative motion between the camera and part or all of the scene. Such unmatchabilities cannot be detected by examining either picture alone, but are discovered only after the expense of attempting to match has been incurred. Since, in this case, the target area has no proper match, the candidate area having the highest correlation will be an incorrect match. It is desirable to be able to detect these incorrect matchings as they occur.

If two areas do not match, the correlation between them should be low. It seems reasonable, therefore, to detect bad matches by seeing if the best correlation obtained was too low. Matters are complicated by the fact that some good matches have low correlations. In fact, for almost any pair of pictures and fixed threshold, it is possible to find either a target area for which there is a bad match with a correlation above that threshold or a target area whose proper match has a correlation below the threshold.

So, how does one distinguish between good matches with low correlations and bad matches? As previously stated, the correlation peak for a proper match should very closely resemble the autocorrelation peak for the target area. In particular, if we have restricted our target areas to those with distinct autocorrelation peaks, a flat or chaotic correlation peak is an indication of a questionable match.

The fact that the correlation and autocorrelation peaks should be similar can be used to derive an autocorrelation threshold for the match correlation. By examining the autocorrelation surface at points near the summit of the autocorrelation peak, it is possible to predict what the correlation should be. (See Appendix B.) Any match below this autocorrelation threshold is highly suspect.

Of course, global information, such as continuity from neighboring points can also be used to determine the credibility of a match.

NON-UNIQUE MATCHINGS

A related problem is that of multiple matches. Since we have not specifically limited the subject matter of our pictures, it is possible that more than one of some object can appear in the pictures. If several of these objects appear against similar backgrounds, a target area can quite reasonably have not one but several matches.

If several areas match the target area, they can be expected to all have about the same correlation. If they are good matches, all of them should be greater than the autocorrelation threshold for the target area. Therefore, to detect multiple matches, one checks to see if there is more than one correlation above the autocorrelation threshold. If so, one checks how well they group. If the top few correlations above the autocorrelation threshold are roughly the same in value, multiple matches are indicated. This can be confirmed by checking to see if the multiple candidates correlate well with each other.

If only the information in the areas is present, an area with more than one match indicated is no more useful than an area with no match indicated, since in neither case has the location of the match been determined. Additional, more global information in the form of a camera model or other matches to tie to can be used to resolve the ambiguity.

WHAT TO DO WHEN A TARGET WON'T MATCH PROPERLY

For some of the target areas which won't match properly using measure of match techniques, there is nothing that can be done. Target areas whose matches fall out of the field of view of the second camera are clearly in this class. Target areas of low information cannot be matched reliably, therefore are assigned to this class. Target areas containing distortion due to perspective change by definition do not have matches, therefore are also assigned to this class. The only reasonable thing to do with targets of these varieties is to give up on them.

Other types of unmatched target areas may be matchable by some different algorithm, probably utilizing more global information. If, for instance, we are employing the similarity heuristic, and it fails for some reason, it may be that pure gridding will find the match. Ambiguous matches and linear edges between areas of low information (which can be thought of as extended ambiguities) can usually be resolved by algorithms which employ additional information, such as a good camera model.

Having a camera model enables one to find the line segment in the second image which corresponds to the center point of the target area. If one of the proposed candidate areas has a center point that lies within one pixel of this line segment, then the match is resolved. This algorithm fails if more than one proposed candidate lies within one pixel of the magic line segment, ie. if two or more of the nominated objects are approximately coplanar with the two camera principal points. This is a fairly common occurrence, since a man-made world containing identical objects is likely to have these objects on a flat surface.

The presence of a set of other matches can also be used to resolve ambiguities. The target area will have some spatial relationship to the target areas of the set; the match is the proposed candidate which most closely approximates this relationship with the candidate areas of the match set [Fischler and Elschlager, 1971]. Of course, care must be exercised in the choice of the set of points. If, for instance, one's anchor points are all in the foreground in the barn pictures, and one is trying to match the fence posts across the field, one will get a meaningless answer. The anchor point pairs used should be at the same depth as the target area to guarantee correct results.

In the case of depth discontinuities, one could employ edge techniques [Hueckel, 1969] to segment the target area into regions. These irregular areas could then have matching attempted on each of them separately, using masked correlation or pointer correlation. (See Appendix B.)

Various methods exist for handling individual unmatchable target areas. In each case, it is first necessary to determine which variety of unmatchability one has, then apply the proper method. Quite often, this is done by the experimenter peeking; that is, the experimenter figures out what kind of unmatchability he has and tells the "algorithm" what to do.

This author has found, however, that the best thing to do with an unmatchable target area is to give up on it and try a different target area. Eventually, target areas that have good matches will come along. (If not, the experimenter SHOULD peek to see if he has the right two pictures!) With good matches, the technique of region growing becomes applicable. Most of the problems related to unmatchable areas can be solved or greatly simplified by the use of region growing.

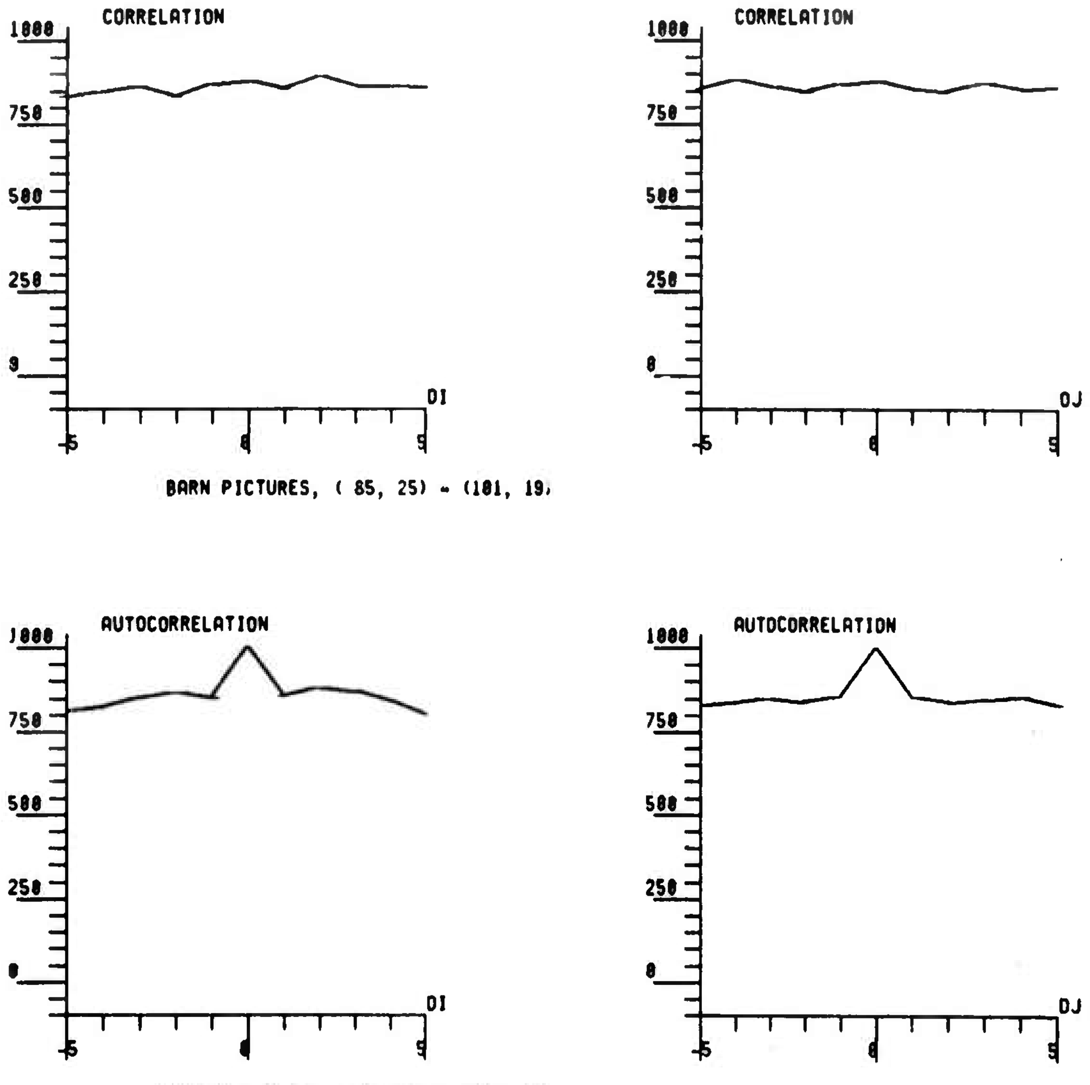


Illustration 4-1. Correlation and autocorrelation graphs for an area of low information, showing the two-dimensional flatness of such peaks.

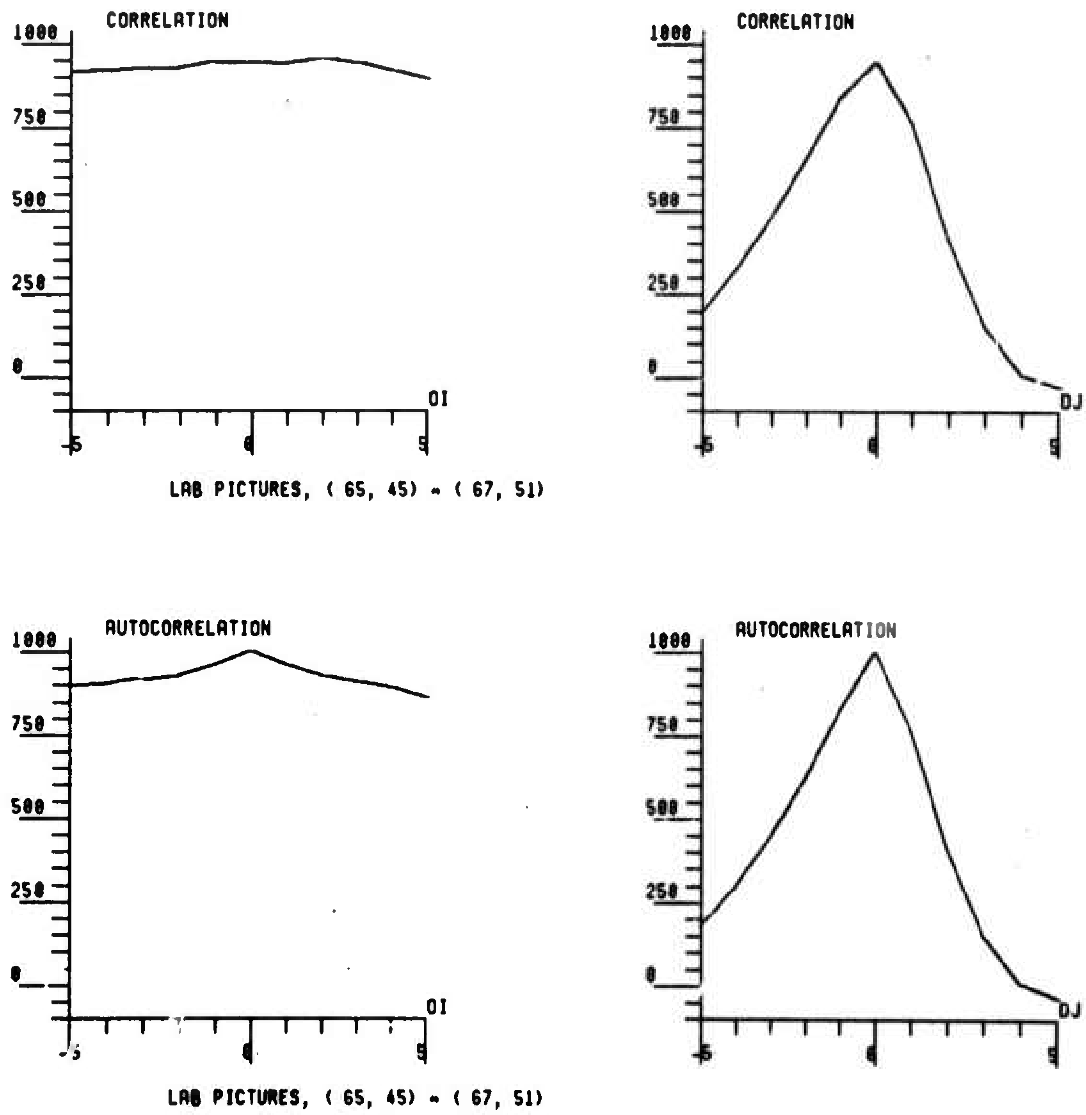


Illustration 4-2. Correlation and autocorrelation graphs for an area with a strong linear edge, showing the one-dimensional flatness of such peaks.

Chapter 5

EXTENDING MATCHES

In Chapter 3, we mentioned the continuity assumption as a crude form of world model which greatly shortened most searches for a match when there was an adjacent match available. This continuity assumption forms the basis for the technique of extending matches.

REGION GROWING: THE BASIC TECHNIQUE

Under the continuity assumption, if the target area centered at (Ix,Jx) matches the candidate area centered at (Iy,Jy), then one would expect the four adjacent target areas (Ix+1,Jx), (Ix-1,Jx), (Ix,Jx+1), and (Ix,Jx-1) to match the four adjacent candidate areas (Iy+1,Jy), (Iy-1,Jy) (Iy,Jy+1), and (Iy,Jy-1), respectively. If (Ix,Jx) matches (Iy,Jy), then the correlation between these two areas represents the peak of the correlation surface and is greater than the autocorrelation threshold for (Ix,Jx) mentioned in Chapter 4 and described more thoroughly in Appendix B. If the four adjacent expected matches are indeed matches, then each of them should meet this same criterion. Once one of the expected matches meets the criterion, then the paired areas adjacent to it become expected matches, etc., and a region of constant (dI,dJ) = (Iy,Jy) - (Ix,Jx) is grown.

Expressed more formally, given a criterion for judging whether or not a point belongs within a region and at least one point at which that criterion is met, the following algorithm extends the region.

1. Push onto the stack at least one point which meets the criterion.

2. Pop one point off of the stack and examine the points lying above, below, right, and left of it. Examining a point consists of first checking to see if it is marked as having been processed; if so, nothing further is done to it. Otherwise, if it meets the criterion of the region being grown, then it is marked GOOD and pushed onto the stack, else it is marked BAD and not pushed.

3. Continue step 2 until the stack is empty.

Marking the points not only leaves behind a record of which are good and which are bad matches, but also keeps the algorithm from repeating work which has already been done. Since there are only a finite number of points available to try, this avoidance of repeated work guarantees that the algorithm will terminate.

EXPEDITING REGION GROWING

As its criterion for a match having occurred, the preceding algorithm uses the fact that we are at a correlation peak and that the maximum correlation is greater than the autocorrelation threshold. For each match pair, this requires ten correlations--nine to determine if the expected match is indeed a correlation peak and one to calculate the autocorrelation threshold.

In practice, eight of the nine correlations are not usually needed. The autocorrelation threshold is derived from expected values of the correlation surface at one pixel displacement from the match. In most caees, the actual correlation at one pixel displacement ie lower than the expected correlation at that displacement, so that the only part of the correlation surface which lies above the autocorrelation threshold is the match peak itself. Testing to see that the correlation is greater than the autocorrelation threshold is usually a sufficient criterion for determining whether or not the expected match is indeed a match.

The correlation between the proposed matching areas and the autocorrelation threshold for the target area still need to be calculated. These two measures each require covering the target area while forming sums. If the sums for both measures are calculated together in one pass over the data, the target area need only be covered once, rather than twice. Thus the combination of the correlation and autocorrelation will take about three-quarters of the time necessary for calculating both separately, or approximately 1.5 times as long as an ordinary correlation.

This is effectively the optimum technique for determining a match. It requires only 1.5 correlations, as opposed to N^2 correlations for the direct method, a savings of a factor of N^2.

EXTENDING MATCHING REGIONS

In our revised algorithm, an area center would be marked BAD if its correlation were not greater than its autocorrelation. For such pointe, the pair of areas may or may not represent a correlation peak.

If the pair of areas does not represent a correlation peak, the continuity assumption need not have been violated. It could well be that this particular part of the scene is continuous, but that the normal to the surface is at a moderate angle to the camera principal axes. Thie can cause a gradual change in (dI,dJ) as one moves across the picture. If this ie the case, then a ehort local search should reveal the correlation peak which represente the match. For this purpose, ueing one "loop" of the epiraling eearch eubroutine MATCH, described in Appendix B, worke quite well.

Once the peak is found, it may or may not pass the autocorrelation threshold. If it does, then this new pair of (Ix,Jx) and (Iy,Jy) becomes a candidate for the application of the region growing algorithm, and the region continues to expand. Illustration 5-1 shows one of the results of this extended region grower.

Any pair of areas that represents a correlation peak but does not pass the autocorrelation test remains unmatched for the present, since in theory that target area has a match elsewhere, which a later region growing will locate.

HOW REGION GROWING SOLVES THE PROBLEMS

In Chapter 4, we promised that region growing would solve, or at least simplify, most of the problems encountered in matching. We divided the unmatchable areas into two categories--those, such as ambiguities and depth discontinuities, which could be matched or partially matched by special means and those which simply had no match, whether due to obscurations, distortions, or changes in the field of view. The problem was that, except for ambiguities, we had no way of telling which variety of unmatchability a given target area might be. If a given target wouldn't match, "peeking" was the only way of telling whether the area was a depth discontinuity which should be segmented or an obscuration which should have no further time wasted on it. Region growing from a few good matches spread about the picture helps here.

Suppose, for instance, a target area which previously failed to match now falls within a region of grown matches. If the target failed to match because of an ambiguity, whether one caused by multiple objects or a linear edge, this ambiguity has been resolved. If the target area didn't match because of a failure of the heuristics, the difficulty has now been surpassed.

Suppose the unmatched target lies just outside of a grown region. If target areas leading up to the unmatched target should match candidate areas leading to the edge of the image, then the intuitive match for our unmatched target area falls out of the field of view of the second camera. In a similar fashion, an unmatched target whose intuitive match has been obscured can now be detected; target areas leading up to the unmatched target will match candidates that lead into a region of candidates having a different matching (dI,dJ)--that of the obscuring object.

If the unmatched target lies in the midst of a "hole" in a grown region, then a moving object which has disappeared, such as the man on the steps in the lab pictures, is indicated. If the unmatched target lies near the edges of two grown regions with rather different matching (dI,dJ), then chances are that the unmatched target contains the depth discontinuity between these two regions.

For most parts of most allowable pairs of stereo images, the continuity assumption holds, so region growing can usually match almost all of the areas of most pairs given just a few "starter" matches. For example, all of the matchable area of the lab pictures can be grown from one match in the background; in the canyon pictures, three matches are required--one on the background canyon wall, one on the foreground promontory, and one on the pinnacle at the right.

Because of the area-based nature of matching, region growing stops when the area reaches a depth discontinuity or touches a distorted region. In the finished products, such as Illustration 5-1, what is displayed is the outer line of center points which the region grower found not to match. Consequently, these products do not precisely outline depth discontinuities or areas of distortion, but fall W pixels away from these edges, where W is the area radius. However, if one is willing to iterate around the edges using smaller and smaller values of W, then closer and closer approximations of these outlines can be found [Levine, 1973].

Thus we see that region growing not only makes it easy to distinguish what type of unmatchability one has, but also does what matching or partial matching is needed. This is why we claimed that region growing would solve or simplify all of the problems attendant to unmatchabilities.

GROWING UNIFORM REGIONS

Indeed, match extension region growing helps with all of the unmatchable areas save those due to low information. As we noted in Chapter 4, areas of low information tend to be areas of low variance. Once such an area has been located in the first image, the technique of region growing can be used to mark that region so that future attempts at matches can be forewarned of the condition.

For this application, the region growing algorithm presented in this chapter need only be modified slightly. As its criterion for a good point, the uniform region grower will use the fact that the variance over the area centered at that point is below a given threshold. Thus instead of comparing areas out of two images and continuing growth if they match, we are evaluating an area in a single picture and growing if that area is of low variance.

As Illustration 5-2a shows, uniform regions grown by this method will stop a bit short of their edges, since any point whose area touches the edge will have a higher variance, thus be rejected. Whether this is bad or good depends on whether the user wanted to delimit the entire uniform region or only that part of it which had too little information to match upon.

If the desired effect was that of Illustration 5-2b then a somewhat different criterion needs to be employed. Low variance means that the average squared difference between the intensity at a pixel and the mean intensity over the area is small. For an area to have a small variance, most of these differences at individual pixels must be small. Hence, we substitute into the uniform region growing algorithm the criterion that the absolute difference between the intensity at a point and the mean intensity over the uniform region be small.

Whether the mean intensity is taken over all of the region grown so far or only over a local part of the region depends on whether the user wishes the uniform region grower to stick strictly to a particular intensity or allow it to follow shading. or to allow it to follow gradual changes in intensity or color, such as occur in a clear summer sky. How small the absolute difference in intensities must be at each point is based on how much variation is expected (or desired) within the area to be grown, and can either be a constant or a statistical measure, such as a multiple of the standard deviation of the intensities within the area. Which uniform region grower one uses, of course, depends upon the effect which the user wishes to produce.

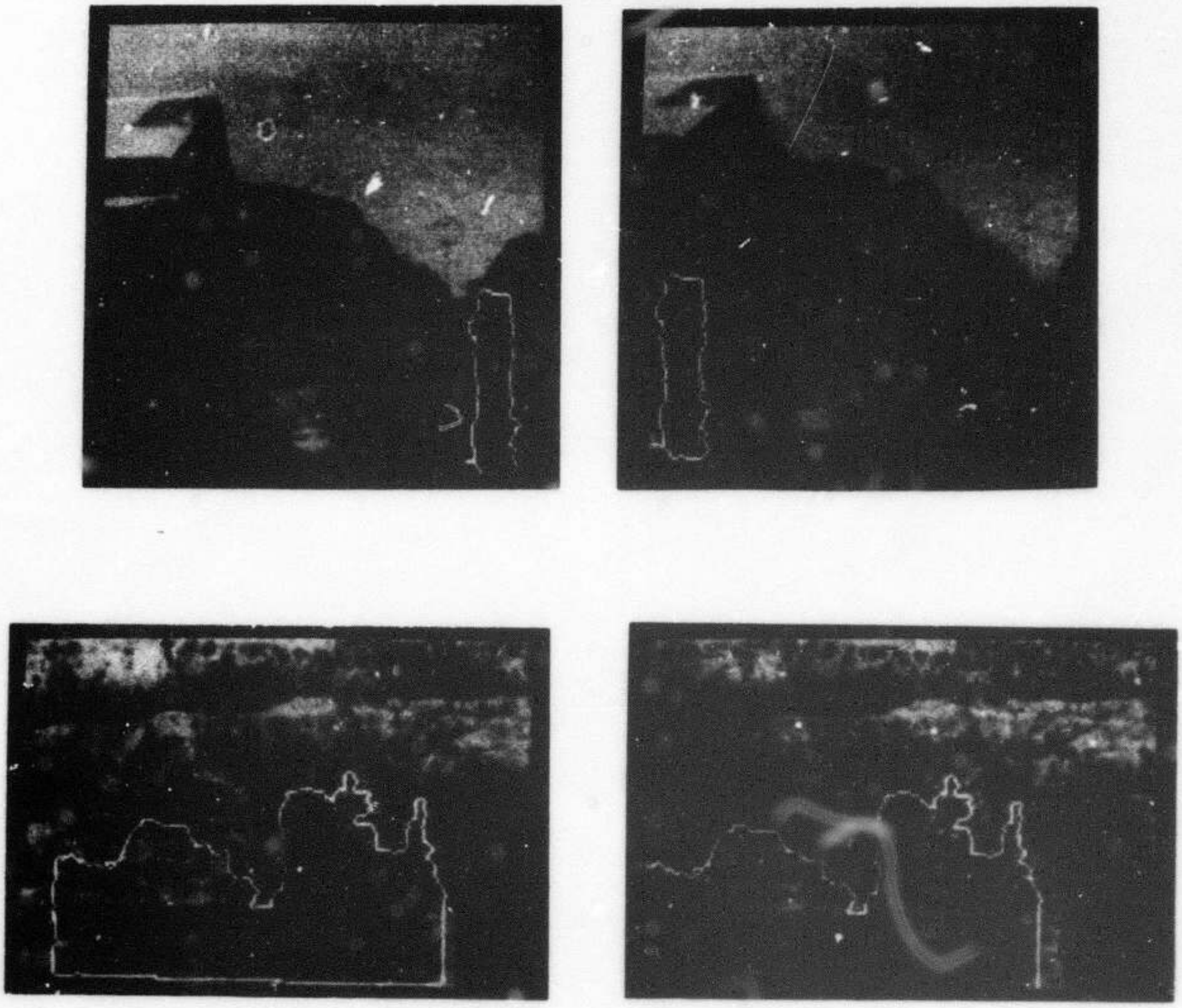


Illustration 5-1. Two pairs of pictures with overlays to show regions delimited by the extended region grower. In the barn pair, the foreground post has been outlined; in the canyon pair the nearest spine of the foreground promontory is shown. Each of these regions consists of several sub-regions at slightly different displacements.

Illustration 5-2. Uniform regions delimited by the region grower. Part (a) shows regions grown by the variance-over-a-window method; part (b) shows the same regions grown by the deviation-from-the-mean method.

Chapter 6

ALGORITHMS AND EXAMPLES

So far, we have presented a variety of techniques, mentioning only briefly how they might be used. In this chapter, we discuss algorithms which use these techniques and give examples of their results.

INDIVIDUAL MATCHES

Sets of individual matches can be used for a variety of things. They can be used to align data for further processing such as differencing [Quam, 1971]. They can be used to derive camera models (see Appendix C). With a camera model, a pair of matching points can be used to determine the relative depth to an object in a scene (see Appendix C). Matches and a camera model make it possible to create a 3-dimensional world model [Baumgart, unpublished research, 1973].

For most applications, there is no need to match particular areas. What is needed is a set of matches that are well distributed in both images. Since very precise matches are usually needed for modelling work, it will be necessary to interpolate discrete matches in order to determine the exact translation. (See Appendix B for a discussion of the need for and techniques of interpolation.) Whenever possible, one should choose the target areas so that matching will be easy and interpolation will be accurate.

Choosing a Target Area

Interpolation is most accurate if the match peak is well behaved--not too flat, not too sharply peaked, and definitely not multi-modal. Since the correlation peak should closely resemble the autocorrelation peak, target areas should be limited to those with well behaved autocorrelation peaks. The target areas whose autocorrelation peaks can be easily fitted by a bivariate normal distribution surface are most likely to yield accurate interpolated match displacements.

Requiring well behaved autocorrelation peaks will also exclude targets which will be hard to match. Flat autocorrelation peaks due to low information, sharp peaks due to only high frequency information being present, and multi-modal peaks due to ambiguities will all be avoided.

To make matching easy, target areas should first of all contain sufficient information. Therefore, only areas having a variance above some threshold should be considered. A reasonable strategy is to first match those target areas that have the highest variance. Of course, high variance can indicate the presence of sharp edges, so each such target area should be checked to see that it is not crossed by a strong linear edge between two low variance areas.

If similarity is to be employed in matching, a quick perusal of the vectors for the representative areas in the second image can be informative. For instance, if the second image contains lots of green areas, but only a few red ones, then one can get some matches cheaply by first matching target areas with red in them.

Program Outline

A program which is to produce a set of well distributed good matches might proceed as follows.

INITIALIZATION. First of all, reduce both images and divide them into representative areas the size of the correlation windows to be used. (Unless otherwise stated, all of the steps that follow are to be carried out in the reduced pictures.) The areas in the first image may simply cover the picture; those in the second image should be on a finer grid so that they overlap significantly. (See Illustration 6-1) Then calculate the vectors of statistics for these representative areas. Histogram each of the components of the vectors for each picture.

RANK TARGET AREAS. Now, do any of the component histograms show only a few targets areas having some property (like being red)? Do at least that number of candidates show that property (if not, some of the target areas will be out of the field of view in the second image, hence unmatchable). Put any areas which seem likely to be easy to match at the head of a list of target areas to be tried.

Next, sort the remaining target areas by their variance. Place those with variances above the low information threshold on the list. Also sort the candidate areas by variance and remove any with too low variance. Since we have removed the low variance target areas, it is unlikely that any of the low variance candidate areas will be needed. Start matching targets off the top of the list.

TARGET MATCHING. For each target area, check to see if its autocorrelation surface is well behaved. If so, establish the autocorrelation threshold and grid spacing parameters for that target area and continue. If not, discard the area and try the next one.

Calculate the similarity measure between the target area and each of the remaining candidate areas and sort them. Start on the most likely area.

Using the grid spacing established for that target, grid the candidate areas and look for a correlation above the noise threshold. Then search the immediate neighborhood for the best correlation (or simply employ MATCH, described in Appendix B).

If ambiguous matches are not anticipated to be a problem, stop examining candidate areas as soon as a candidate is found that has a correlation above the target area's autocorrelation threshold. Otherwise, continue examining areas until the measure of similarity becomes too dissimilar. If no candidate had a correlation that was high enough, forget that target area.

Now go back into the original, full resolution pictures. Re-determine the autocorrelation threshold for the full resolution target area. Re-optimize the correlation for each of the promising candidate areas. Test these correlations for bad matches and ambiguity. Discard the target area if it fails these tests, otherwise interpolate the match in the full resolution pictures and record it. Go on to the next target area.

Continue matching target areas in this fashion until enough matches with the proper spatial distribution are obtained or the list of matchable areas is exhausted. Take the results and do your thing with them.

The algorithm described here has not been implemented in totality, however, most of its pieces have been implemented. Reduction of images is accomplished by a program called PICSEE by Lynn Quam. The initialization and sorting of target areas is done by the author's program VECTDO which calculates the color vectors described in Chapter 3 or VECTBO which does the black-and-white vectors. The target matching is done by the author's program NEWPTS. Final decisions in the full-scale images are done by the author's program REFINE. All of these programs are written in the SAIL dialect of ALGOL [VanLehn, 1973] at the Stanford Artificial Intelligence Project. Critical inner loops are written in START_CODE, an embedding of PDP-10 assembly language into SAIL.

Illustration 6-2 shows a set of matches produced by this system of programs and run through the author's program DEPTH to figure the depthe at each point pair in meters.

A COMPLETE MATCHING

The ultimate combination of matching techniques occurs in an algorithm for creating a complete matching. Such an algorithm puts together all of the techniques we have developed and shows how they interrelate.

We begin with the algorithm described in the first section of thie chapter. This gives us a set of precise interpolated matching areas. We feed the point pairs to a camera model derivation routine which returns a camera model.

Next we seek low variance regions and employ one of the uniform region growers described in Chapter 5 to color these regions unmatchable. All region growing is done in an auxiliary "picture" which we will use to keep track of the parts of the first image that we have processed and to record the matches which have been made.

The matches which determined the camera model are then un-interpolated--that is, they revert to the discrete form they had before interpolation--and put onto a stack of regions to be extended. A match pair is popped off of this stack and passed to the region grower for extending matches. As the region grower proceeds, it marks in the areas it grows in the recording picture and in a second auxiliary picture which keeps track of which area centers in the second image have been matched.

When the region grower finishes each sub-region having the same displacement (dI,dJ), a cleanup algorithm goes around to all of the points marked BAD on that round. So that future growings can have a chance to work on them, they are re-marked as being unmatched and placed on the stack of pairs of points waiting to have the region grower applied.

Each pair of points taken off of this stack is re-MATCHed (see Appendix B) to find the correlation peak, which is compared to the autocorrelation threshold. Point pairs which pass this criterion, and haven't been overgrown by some previous extension, are passed to the region grower, until the stack of point pairs awaiting the region grower becomes empty.

When the original set of matches has been exhausted, we begin looking in the recording picture for areas which have not been marked. For a representative point in the midst of such a region, we attempt matching using the camera model as described in Chapter 3. In this case, we can further limit our search along the back-projected line in the second image with the knowledge that some of the points in the second image have already been matched, hence do not need to be considered. For each match found this way, the region grower is started up again. This continues until all of the unmatched areas have either been examined or are smaller than some critical size, below which we do not bother with them.

As yet, this algorithm has not been implemented as a whole. However, most of the parts do exist as separate programs which communicate with each other via disk files of data. In addition to the programs described in the last section for finding a set of well-distributed matches, we use CAMERA, written by Donald Gennery at Stanford A.I. to determine the camera model corresponding to our set of point pairs. The finding and marking of low variance areas is done by the author's program LOWINF. The actual extension of regions from a file of matching area centers is done by the author's program MGROW. The camera model search for matching point paire is implemented by the author's program CAMSCH. As with the programs from the last section, these were written in SAIL on the PDP-10 at the Stanford Artificial Intelligence Laboratory.

Illustration 6-3 shows the results of the author's program EMAKE on a complete mapping generated by this system of programs.

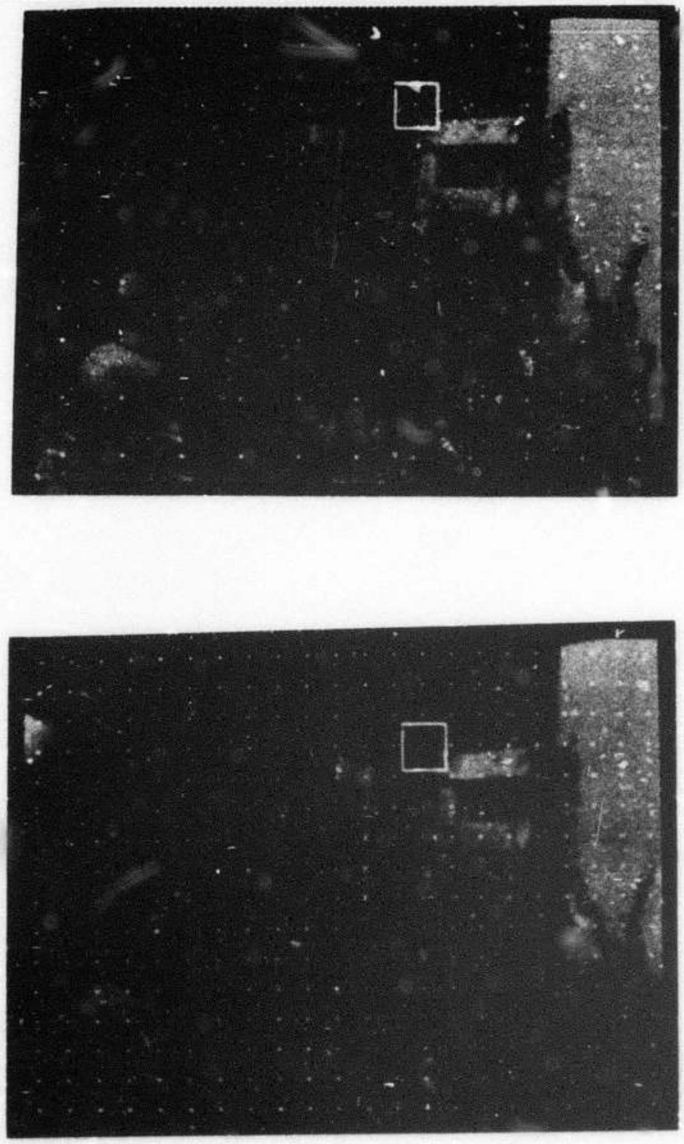
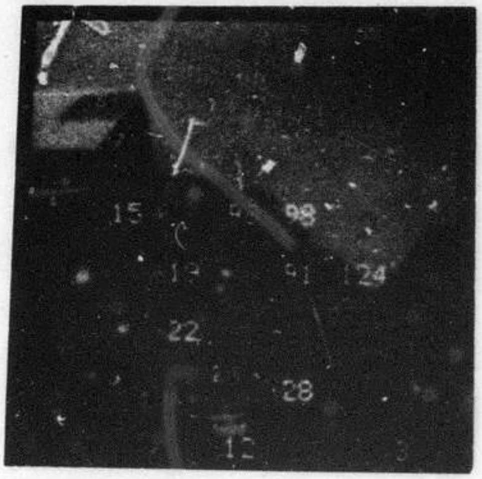


Illustration 6-1. Yard pictures, with overlaid grids for target and candidate areas. Notice that the candidate areas are on a much finer grid than the target areas. A typical target and the representative candidate which most closely resembles it are indicated by squares.



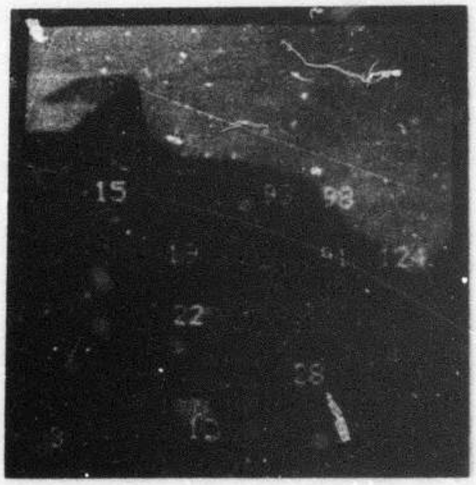


Illustration 6-2. Barn pictures, showing a set of matches produced by NEWPTS. The dots indicate the center points of the matching areas; the numbers by the dots give the distances in meters from the first camera to the 3-dimensional points which correspond to the point pairs.

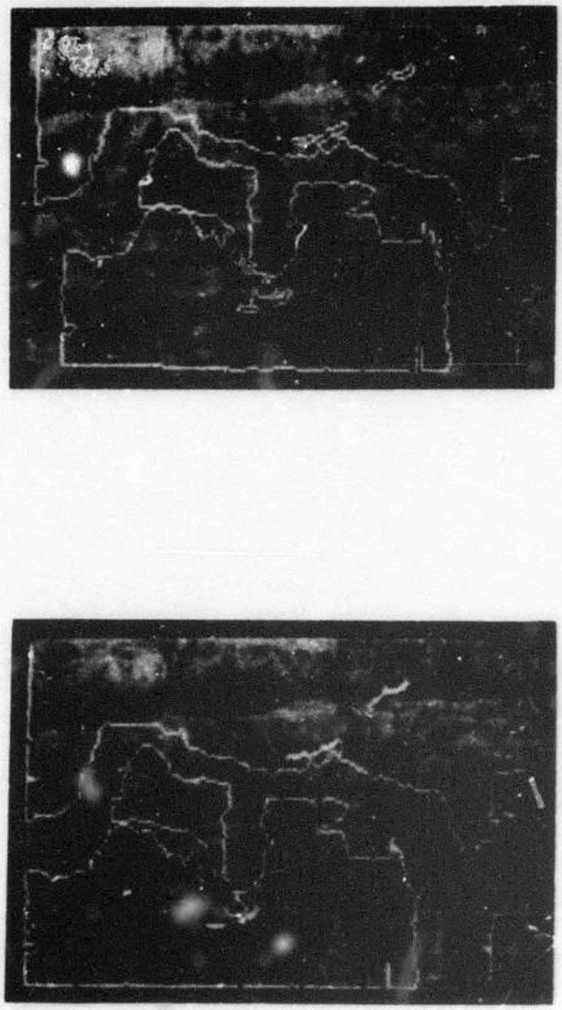


Illustration 6-3. The canyon pictures, showing a complete matching. The outlines show major depth discontinuities and delimit areas which could not be matched.

Chapter 7

CONCLUSION

It was the purpose of this thesis to investigate techniques by which areas of one picture could be matched with the corresponding areas from the second image of a stereo pair. We started with the assumption that we had two images of the same scene which differed somewhat, but the majority of which could be matched (as opposed to mapped, which is a different processe). That is, we treated those parts of the scene for which no gross dietortione had been introduced between the two views. Our objective of making matches efficiently (ie. without calculating the correlation between the target area and candidate areas centered at every point in the second picture), was to be reached by presenting techniques by which this could be accomplished.

ACHIEVEMENTS

In this thesis, we have presented tools and techniques by which areae in one picture can be efficiently matched with the corresponding areas in the second picture.

We have discussed three measures of match which are suitable for thie purpose, normalized cross-correlation, root-mean-square error and absolute difference. In addition to the ordinary one dimensional versions of these measures, we have documented correlation for use in two dimensions, derived color or vector correlation, masked correlation, and weighted correlation, and explained function correlation, which can be used for mapping. We have discussed some properties and relative efficiencies of the baeic measures. We have mentioned the existing techniques of fast Fourier convolution and sampling for making the calculation of these basic measures more efficient, but pointed out their shortcomings. It is our position that our techniquee have none of these shortcomings and are more efficient that theee other methods.

We have discussed several methods for pruning the search for a match. Gridding and reduction each give a savings factor of n^2, where n depende on the data in the images, but is typically 3 (savings factor is 9) for gridding and 5 to 10 (savings factor is 25 to 100) for reduction. Similarity gives a savings factor of 100 to 150 for the author's data. Camera models give a savings factor of N, the width of the picture--typically about 200. World model assumptions can result in a savings factor of almost N^2, the area of the picture--typically 10,000 to 40,000.

For those who do not have camera models given, we have included the mathematics necessary to convert a set of matchings into a workable camera model. We have also included calculations which use this model to find the depth of the 3-dimensional point corresponding to a given pair of image points.

We have discussed the fact that, with real data, not all target areas are matchable. We have given methods by which some of the major typee of these unmatchabilities can be detected in the original data. Since some unmatchable targets cannot be detected directly, we have developed methods for detecting when a proposed match is not really a match.

We have discussed region growing techniques which can be used to extend matching areas. Because these are based on the continuity assumption, a sort of low level world model assumption, they are quite efficient methods of finding matches. We have also presented region growing techniques which can be employed to delimit uniform regions in one image.

Finally, we have presented two algorithms demonstrating how the abetract techniques we have developed and documented can be combined to perform useful functions in the processing of stereo images.

APPLICATIONS

Some of the techniques of this thesis have already been adapted for use in various artificial intelligence and robotics tasks. In addition to the author's programs mentioned in Chapter 6, the reduction, gridding, and eimilarity techniques and the uniform region growing have been incorporated into programs for servo-ing a computer driven cart [Quam, undocumented research, 1973]. Gridding and the continuity assumption form the basis for programs in a feasiblity study for automating photogrammetric etudies of the planet Mars during the 1975 Viking mission [Quam, unpublished research proposal, 1973]. The complete matching techniques described in Chapter 6 will undoubtedly play a part in this application, also.

Applications of multiple image processing also occur in medical research. The registration of time-lapse x-rays for further processing is only one of many possibilities.

Another eventual application is planetary exploration. For inhospitable environments and extreme distances, on-board computer processing of images will be vital to mission success.

AREAS FOR FURTHER INVESTIGATION

In the process of our investigations, we have discovered a number of areas which need more work, as well as several interesting extensions of our work.

The field of area mapping is for the most part untouched. We have scratched the surface with this thesis on matching and our brief comments in Appendix D. Much more can and should be done in this field. Complete, separate investigations of techniques for motion and near-field stereo are needed.

We have excluded noise from our data. There needs to be extensive work on the effects of noise on matching. Also in need of exploration are the techniques for alignment of regions by boundary matching, touched upon in Appendix D.

We leave these as challenges to future investigators.

Appendix A

THE IMAGES

The techniques and algorithms described in this thesis have been developed and tested using principally four pairs of pictures, which are described and presented in this section. Other pairs of pictures have had isolated techniques used on them, but not sufficiently to warrant their being presented here.

The images used were mainly of outdoor scenes. Some contained man-made objects while others did not. The main criterion for selecting these particular pictures to work with was that they were available and that they had a certain esthetic appeal to the author.

Due to the limited facilities available for printing this thesis, it is not feasible to reproduce the images in color. Consequently, the illustrations presented here are black and white versions of the images used.

THE BARN PICTURES

The first and most used pair of pictures is of a barn and field near the author's home. The barn, a rather rustic building of unpainted wood with a tin roof, appears at the left of the picture. In the foreground is a stock fence of woven wire topped by 3 strands of barbed wire hung on hand-split fence posts. Due to the relative camera motion between the two images of the stereo pair, one of these fenceposts appears at the right of the first image and at the left of the second.

The area in front of the barn is covered with green grass, on which rest several abandoned objects, including two barrels, a lawn chair, and a bench lying on its side. The shadow of a tree behind the camera and to the right falls diagonally across this grassy area.

The grassy area extends into the distance. It is crossed by several fences, one of boards near the barn and the rest of the same materials as the foreground fence. The land rises somewhat; the skyline is a ridge about 120 meters from the camera positions. Two groves of oak trees cover most of this ridge. A telephone pole stands in the small open area on the skyline between the two groves.

The original photographs were 35 mm color slides. The cameras were hand held in the field; the distance between the two camera positions is slightly over one meter. The slides were photographed under standard red, green, and blue filters to produce black and white negatives, which were then digitized commercially. The resulting 800 by 1200 pixels of data were windowed to remove a light leak in the lower portion of the foreground fence and spatially reduced by a factor of five to produce 150 by 150 images. Illustration A-1 shows the intensity pictures, made by averaging the red, green, and blue component pictures.

The colors in the picture are mostly blues, greens, and browns. The sky is a clear, saturated blue; the tin on the roof has a blue tinge. The trees and the foreground grass are green. The barn and fence posts are a rusty brown, while the grass in the distance is yellowish brown.

The barn pictures have been used as both color and intensity images. They are the most referred to images in this thesis, partly because they were the first images tried, partly because they present so many different problems and exercises for matching, and partly because they are the author's favorites among the images used.

Actually, the barn pictures violate the hypothesis that the change in point of view does not significantly change the perspective of the scene. The barn door is half-again as wide in the second image as it is in the first, a significant change. These changes, along with the "moved" foreground post, are what make this pair of pictures difficult, hence valuable.

THE LAB PICTURES

The second pair of pictures is of Stanford University's D. C. Powers Laboratory, where the Artificial Intelligence Project is housed, and where the author works. The laboratory building crosses the picture in the middle distance. Behind it is a row of eucalyptus trees, through which the skyline, a ridge about five miles away, can be seen.

The immediate foreground is a roadway. Between the road and the lab building is a parking lot filled with a variety of cars. A grassy area is immediately in front of the building, divided by a concrete walk with steps. A few cars are parked on this grassy area slightly left of the center of the images. Due to the slight time difference between the actual taking of the two photographs, there is a man walking down the steps in the first picture who does not appear in the second picture. Also, one parking space has been emptied and another filled in that time interval.

Lighting is from overhead, with the sun slightly in front of the camera. Thus the near faces of the building, cars, and even the trees are in shadow. Some reflection occurs from automobile windshields. Since the day was slightly smoggy, shadows are slightly diffuse and the distant hills hardly visible.

The original photographs were 35 mm color slides. The camerae were hand held in the field; the distance between the two camera positions is approximately ten meters. The slides were photographed under standard red, green, and blue filters to produce black and white negatives, which were then digitized commercially. The resulting 1200 by 800 pixels of data were windowed to remove a light leak at the right end of the building and spatially reduced by a factor of five to produce 150 by 150 images. Illustration A-2 shows the intensity pictures, made by averaging the red, green, and blue component pictures.

Colors are predominantly blues and yellows. The shadows on the trees and building override their true colors with a blue tinge. Most of the cars in the lot are blue, grey, or white; the station wagon in the first row is red, but again its color is largely masked by the shadowed near side and the glare off of the hood. The grassy areas are yellow, with some green along the walkway.

The lab pictures have been used as both color and intensity images. In spite of the wide separation between the cameras, all of the objects are far enough away to avoid problems with perspective distortion. However, the presence of many man-made objects of uniform color and having linear edges makes this pair of pictures interesting.

THE CANYON PICTURES

The third pair of pictures were taken from the rim in Bryce Canyon National Park of one of their sandstone formations. In the middle distance are pinnacles and a narrow spine of eroded sandstone running across the picture. In the far distance is the other side of the canyon with sparse evergreen trees clinging to it. Lighting is from the right, casting many of the faces of the pinnacles into shadow.

The original photographs were 35 mm color slides. The cameras were hand held in the field; the distance between the two camera positions is approximately fifty meters. The slides were digitized by use of a special illuminating attachment to one of the A.I. Lab Hand-Eye television camerae. The pictures in Illustration A-3 are full scale 180 by 120 windows out of the middle of the originals, to pick up the most challenging features.

This was a particularly interesting pair of pictures from an artificial intelligence point of view. Using only the intensity information, humans were, for the most part, unable to pick out the exact location of the edges of the mid-ground formations. Color information helped, since the background formations are yellow-orange, while the midground ones are more red; also the green of the trees helped to distinguish them from dark shadows in crevasses. Still, some edges required looking at both of the color pictures before people could locate them exactly. The challenge was to see whether the matching and depth discontinuity algorithms could do well with only stereo intensity information.

THE YARD PICTURES

The fourth pair of pictures is of a portion of the area around the author's home. Part of the cinder-block garage wall is visible at the right side of the picture, with ivy growing up it. A board fence extends from the corner of the building across the picture. Pyracantha bushes obscure the fence at the left edge of the picture. The fence is broken in the middle of the picture by a wooden gate, which is standing open, away from the camera. There is a small rug hanging on the gate, a pair of gloves on the fence, and a jar sitting on the gate latch post.

Two large firewood logs are in the foreground in the middle of the picture, one lying on its side and one standing on end. The one on end has an ax handle lying across it; the ax head is embedded in the top of the log. An automobile wheel lies between the upright log and the ivy. There is a plastic dish-pan upside down under the pyracantha nearest the gate. The roof line of another building is visible just over the fence. Tree tops form the background of the picture.

The original photographs were 35 mm black and white negatives. The cameras were hand held in the field; the distance between the two camera positions is approximately one meter. The negatives were digitized commercially, and the 800 by 1200 pixels of data were windowed slightly and spatially reduced by a factor of five to produce 220 by 160 images. Illustration A-4 shows the resulting images.

Again, parts of this pair violate the hypothesis of no perspective distortion. Specifically, the foreground logs and the ax handle show significant differences in orientation in the two images. However, the other parts of the images make excellent material for matching. Like the canyon pictures, humans have some difficulty separating the background from the foreground in this pair, particularly where the pyracantha bushes blend into the background trees.

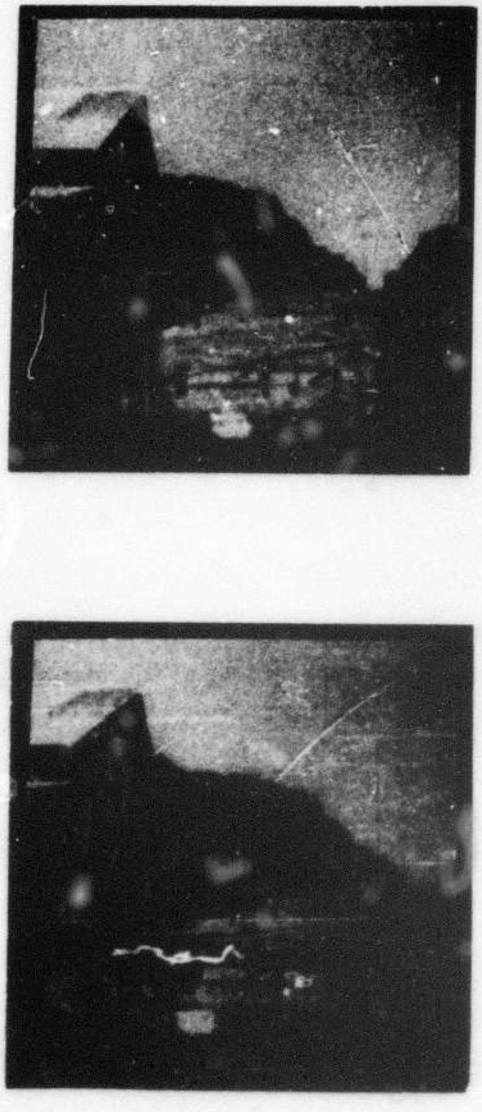


Illustration A-1. The barn pictures.

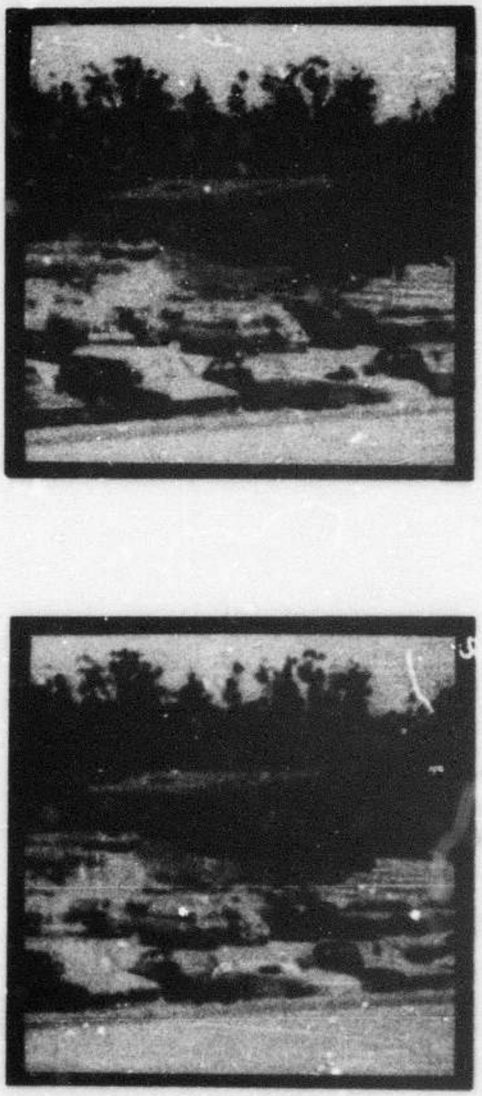


Illustration A-2. The lab pictures.

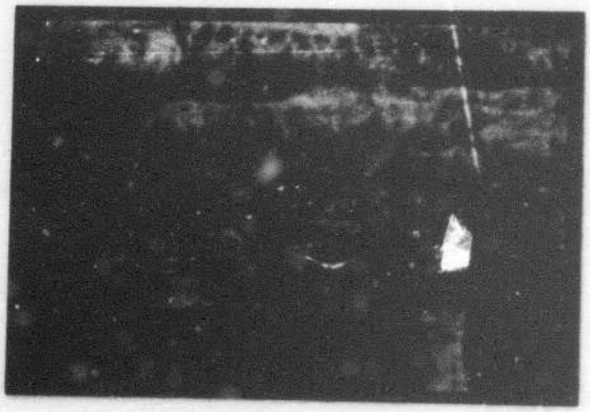


Illustration A-3. The canyon pictures.

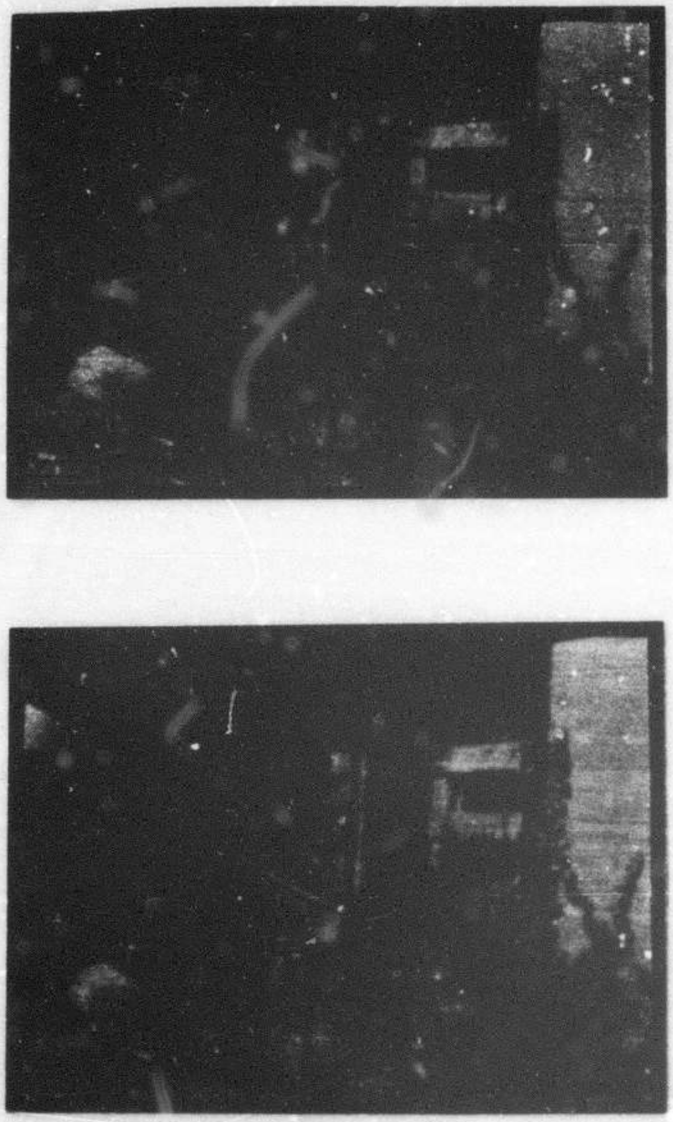


Illustration A-4. The yard pictures.

Appendix B

BASIC CORRELATION TOOLS

For the purposes of this thesis, the measure of match between two areas will be normalized cross-correlation. It will ordinarily be calculated between areas that are rectangular in shape and have odd dimensions, ie. 2m+1 × 2n+1 windows. This makes it easy to characterize the area by its center point.

In this and the following appendices, the following mathematical conventions are used.

Vectors are indicated by an arrow $\vec{}$ over the capital letter which names the vector, e.g. $\vec{A}$ is the vector named "A". Unit vectors are indicated by a hat ^ over the lower case letter which names the vector, e.g. $\hat{a}$ is the unit vector named "a". Specific 2- and 3-dimensional vectors may be written out (x,y) or (x,y,z), respectively.

Vector dot product is indicated by a raised dot •.

The norm or length of a vector $\vec{A}$ is denoted by $|\vec{A}|$.

The mean of a vector quantity $\vec{A}$ is denoted by $\bar{\vec{A}}$.

Exponentials of the quantity e (the base of natural logarithms) are represented by using the function EXP.

AREA CORRELATION

The basic measure of match is the "correlation coefficient" discussed in most elementary statistics books. (For example, see Freund [1962]) In our notation, this correlation is

$$\mathrm{COR} = \frac{\sum_i (X_i - \bar{X}) * (Y_i - \bar{Y})}{\mathrm{SQRT}\left(\sum_i (X_i - \bar{X})^2 * \sum_i (Y_i - \bar{Y})^2 \right)} \qquad \text{(B-a)}$$

For our purposes, X_i and Y_i are intensity values at corresponding pixels within the rectangular windows. This is implemented as

$$COR = C(X, Ix, Jx; Y, Iy, Jy) \qquad \text{(B-b)}$$

$$= \frac{\sum_{-m\le i\le m} \sum_{-n\le j\le n} (X[Ix+i, Jx+j] - \overline{X}) * (Y[Iy+i, Jy+j] - \overline{Y})}{SQRT\left(\left(\sum_{-m\le i\le m} \sum_{-n\le j\le n} (X[Ix+i, Jx+j] - \overline{X})^2 \right) * \left(\sum_{-m\le i\le m} \sum_{-n\le j\le n} (Y[Iy+i, Jy+j] - \overline{Y})^2 \right) \right)}$$

where (Ix,Jx) and (Iy,Jy) are the centers of the target and candidate areae, respectively. Since this is rather cumbersome to write, we will abbreviate it with the notation of Equation B-a, leaving the center points and the fact that i ranges in two dimensions over the (2n+1)*(2m+1) pixele in the surrounding windows implicit. The means, of couree, are calculated over thie same area.

This is our ordinary form of correlation. It ie primarily ueeful in an application where each image consists of one (black-and-white) picture.

COLOR CORRELATION

In the case of color images there are three pictures involved. Since the color images we currently are working with were obtained by digitizing three black and white picturee which resulted from photographing an ordinary color elide under red, green, and blue filters, respectively, we ehall coneider the components of our color picturee to be red, green, and blue, which we will symbolize as R, G, and B.

It ie eomewhat more convenient to think of a color picture P as one array of vector-valued points (PR,PG,PB) inetead of three separate arrays of ecalar-valued points PR, PG, and PB. This suggeets regarding the text-book version of normalized cross-correlation, Equation (B-a), ae the one-dimensional case of a vector function

$$VCOR = \frac{\sum_i (\vec{X}_i - \overline{\vec{X}}) \cdot (\vec{Y}_i - \overline{\vec{Y}})}{SQRT\left(\sum_i | \vec{X}_i - \overline{\vec{X}} |^2 * \sum_i | \vec{Y}_i - \overline{\vec{Y}} |^2 \right)}$$

Coneidering only the numerator of VCOR, and letting $\vec{X}_i$ be $(XR,XG,XB)_i$ and $\vec{Y}_i$ be $(YR,YG,YB)_i$, we have

$$\sum_i (\vec{X}_i - \bar{\vec{X}}) \cdot (\vec{Y}_i - \bar{\vec{Y}})$$

$$= \sum_i ((XR,XG,XB)_i - \overline{(XR,XG,XB)}) \cdot ((YR,YG,YB)_i - \overline{(YR,YG,YB)})$$

$$= \sum_i (XR_i-\overline{XR},\ XG_i-\overline{XG},\ XB_i-\overline{XB}) \cdot (YR_i-\overline{YR},\ YG_i-\overline{YG},\ YB_i-\overline{YB})$$

$$= \sum_i (XR_i-\overline{XR})*(YR_i-\overline{YR}) + (XG_i-\overline{XG})*(YG_i-\overline{YG}) + (XB_i-\overline{XB})*(YB_i-\overline{YB})$$

If we notice that all three terms within this sum are the same in form and change the definition of i so that it ranges over all components as well as all elements of components, we get

$$= \sum_i (X_i - \bar{X}) * (Y_i - \bar{Y})$$

which is the numerator of the formula for ordinary correlation Equation (B-a). By similar manipulations, the two terms in the denominator of VCOR become the same as the two terms in the denominator of Equation (B-a). This means that color correlation is really a dressed up form of ordinary correlation. This is convenient, for it means that color correlation will have all of the properties that ordinary correlation has been observed to have.

MASKED CORRELATION

Obviously correlation need not be restricted to rectangular windows; the correlation coefficient can be calculated over any eample, regardlese of shape. The only reason for using the rectangular windows was that it is easier to set up indices to cover a rectangular area than to make indices trace out an arbitrarily shaped area.

To do correlation over oddly shaped areas, it is first necessary to implement a way of covering arbitrarily shaped areas easily. Toward this end, the idea of a correlation mask has been instituted. The mask consists of a rectangular array M which completely covers the area of interest and is filled with ones in the area of interest, and zeros elsewhere. In effect, M is a template for the irregular area.

To use the mask, one sets up indices to cover the rectangle, as in ordinary correlation, then uses each point of the mask as a predicate to tell whether or not to include the corresponding pixels in the sums for the correlation coefficient. Mathematically, this is equivalent to multiplying each term of the sums by the corresponding term of the mask, that is

$$\mathrm{MCOR} = \frac{\sum\limits_{i|M_i=1} (X_i - \bar{X}) * (Y_i - \bar{Y})}{\mathrm{SQRT}\left(\sum\limits_{i|M_i=1} (X_i - \bar{X})^2 * \sum\limits_{i|M_i=1} (Y_i - \bar{Y})^2 \right)}$$

$$= \frac{\sum\limits_{i} M_i * (X_i - \bar{X}) * (Y_i - \bar{Y})}{\mathrm{SQRT}\left(\sum\limits_{i} M_i * (X_i - \bar{X})^2 * \sum\limits_{i} M_i * (Y_i - \bar{Y})^2 \right)}$$

where it is understood that the summations necessary to calculate the means are done only over the valid part of the mask.

When we attempt to use a zero-one correlation mask to match the top of the foreground fence post in the barn pictures, we discover that the masked post correlates best with a piece of the barn wall below and to the right of the intuitive match. Using the inverse of this correlation mask--keeping the background and masking out the post--works fine; the trees and sky match up as one would intuitively expect them to.

What is the difference between these two cases? In the second case, we are attempting to remove an intruding object and match around it. We don't care what shape the object is; we merely want to get rid of it.

In the first case, we are attempting to match a specific object with definite boundaries. In masking out the background, we have also masked out the fact that the post has edges, turning the post into a piece of wood which matches the wood of the barn as well as it matches its true counterpart in the second image.

In order to match specific objects, it is necessary to somehow retain information about the boundaries of the objects. One way to do this is, rather than masking out everything outside the areas of interest with zeroes, to instead weight the correlation so that all of the window is considered, but the areas of interest influence the correlation more than does their background.

WEIGHTED CORRELATION

This suggests replacing the zero-one correlation mask M by a weight mask W, yielding,

$$WCOR = \frac{\sum_i W_i * (X_i - \bar{X}) * (Y_i - \bar{Y})}{SQRT(\sum_i W_i * (X_i - \bar{X})^2 * \sum_i W_i * (Y_i - \bar{Y})^2)}$$

This necessitates changing the nature of the mean used from the ordinary averaging mean to a weighted mean. Thus, instead of using

$$\bar{X} = \frac{\sum_i X_i}{\sum_i 1}$$

we want to use

$$\bar{X} = \frac{\sum_i W_i * X_i}{\sum_i W_i}$$

Indeed, when the correlation mask for the foreground post is filled with ones and sevens, instead of zeroes and ones, the algorithmic match is the same as the intuitive match: post matches post.

In addition to being used in most template matching, WCOR can also be used to place arbitrary weights in a window, as shown in Illustration B-1.

POINTER CORRELATION

Most correlation is implemented in a very orderly fashion. A pointer starts at the upper left-hand corner of the rectangle to be covered and moves across the row of pixels. When it gets to the right edge of the rectangle, it returns to the left edge in the next row. The reason for this is efficiency.

No matter whether the pixels are placed one per word (or fixed-length byte) or are packed and unpacked by special byte handling instructions, the most efficient way to access an area of bytes is to have a pointer which one increments. The efficiency consideration pretty well constrains one to scanning lines of the picture.

Correlation does not demand this. All that correlation requires is to be given pairs of points, one out of each picture, which are then incorporated into the sums. Another way to implement correlation is to first set up a table of pointers, then simply run a secondary pointer down the table of pointers. Implemented in this fashion, correlation becomes

$$PCOR = \frac{\sum_i (X[P_i] - \bar{X})*(Y[P_i] - \bar{Y})}{SQRT(\sum_i (X[P_i] - \bar{X})^2 * \sum_i (Y[P_i] - \bar{Y})^2)}$$

where i now ranges over the table of pointers, and the means are calculated from this same set of pointed X and Y.

Once one has accepted the extra cost caused by looking up the pointer before one can use it, other benefits become obvious. For instance, we are no longer tied to rectangular areas. Once the pointers are set up, it is immaterial what shape they cover--hexagons, circles, trapezoids, and even grossly irregular shapes are all the same to this correlation. This does away with the need to cover a rectangular template which tells whether or not to include a given point in the correlation. Since as much as half of a template is not used most of the time, not having to consider those points at all could result in a vast speedup of correlating irregular areas.

This form of correlation also makes it possible to correlate in pictures with known distortions. The pointers are simply set up to take the distortion mapping into account. For instance, if one picture is known to have a scale-factor difference from the other, the target area can be covered by pointers at unit spacing while the candidate area is covered by pointers determined by the scale factor. Any other known distortion can be handled similarly.

One can even access the pixels in an area randomly, say to implement a Barnea and Silverman type sampling algorithm. All that is needed are two parallel tables of pointers generated in some pseudo-random order.

AUTOCORRELATION

In signal processing, the autocorrelation function is an important tool for characterizing the frequency content of a signal. The fact that, for suitably constrained signals, the Fourier transform of the autocorrelation function is the power-density spectrum of the signal explains why an examination of the autocorrelation peak can give such a good indication of the presence of extremely high or low frequency components in the image [Lathi, 1968]. Our main interest in autocorrelation, however, is not as a tool for characterizing the image data, but as a tool for determining what correlation values might be expected for a given target area.

Let $A(Ix,Jx;di,dj)$ denote the correlation between an area of picture X centered at (Ix,Jx) and an area of picture X centered at $(Ix+di,Jx+dj)$. In the notation of Equation B-b, this is expressed as

$$A(Ix,Jx;di,dj) = C(\ X,Ix,Jx;\ X,Ix+di,Jx+dj\ )$$

If the two images were identical except for a constant translation (Ti,Tj), gain A, and offset B--ie. $Y[i,j] = A * X[i+Ti,j+Tj]+ B$ for all (i,j) in the images--then the correlation and autocorrelation surfaces would be exactly identical. For a pair of areas centered at (Ix,Jx) and (Iy,Jy) which are an intuitive match, we would have

$$A(Ix,Jx;di,dj) = C(\ X,Ix,Jx;\ Y,Iy+di,Jy+dj\ ) \tag{B-c}$$

for all (di,dj) within the two images.

This is rarely the case, since most data of interest will have more meaningful changes between the two images than a constant translation, gain, and offset. However, when we assumed that there is little or no distortion over windows of the size being correlated, we effectively postulated that the changes between the two images are small locally. Consequently, while Equation B-c usually will not hold for all (di,dj) within the two images, it might be expected to hold within the immediate vicinity of the matching area centers.

Now, we know that correlation of (Ix,Jx) with areas centered at points around (Iy,Jy) yields values not greater than the correlation with (Iy,Jy), ie., for $0<|(di,dj)|<2$,

$$C(\ X,Ix,Jx;\ Y,Iy+di,Jy+dj\ ) \le C(\ X,Ix,Jx;\ Y,Iy,Jy\ ) \tag{B-d}$$

for it was the fact that we were at a correlation peak which helped to determine (Iy,Jy) to be the match. Substituting Equation B-c into the left side of Equation B-d, we have for $0<|(di,dj)|<2$

$$A(Ix,Jx;di,dj) \le C(\ X,Ix,Jx;\ Y,Iy,Jy\ )$$

ie. that the match correlation is not less than any of the immediate neighboring $A(Ix,Jx;di,dj)$. Consequently, we would expect that the correlation is not less than the maximum of these autocorrelations, that is,

$$C(\ X,Ix,Jx;\ Y,Iy,Jy\ ) \ge \max_{0<|(di,dj)|<2} A(Ix,Jx;di,dj)$$

Experimentation has shown that the match correlation meets this criterion for some 90% of the good matches found. In addition, the correlation at false matches fails to meet this criterion for about 95% of the cases examined.

A related measure, an autocorrelation calculated between the target area and a copy of itself created by displacing different parts of the correlation window in different directions as shown in Illustration B-2 also works quite well as a floating threshold. This measure has the advantage that it can be calculated in one pass over the data, rather than the 8 passes required to calculate the 8 neighboring autocorrelations for measures based on $A(Ix,Jx;di,dj)$ for $0<|(di,dj)|<2$. Effectively, this threshold measures how well the target area correlates with a slightly distorted version of itself. A large number of other distortion patterns can also be used.

This autocorrelation threshold passes about 98% of the good matches found, and rejects approximately 99% of the false matches encountered. It is this threshold which is most commonly used in region growing, both because of its ease of calculation and its accuracy of prediction. Unfortunately, we do not know why it seems to function better.

We have discussed autocorrelation in terms of the standard area correlation. Of course, if another form of correlation is used to determine the match, then the autocorrelation must use that same type of correlation, be it masking, weighting, or pointer correlation. Similarly, if the measure of match used is not correlation at all, but one of the difference measures, then the "autocorrelation" becomes the "autodifference". Only the formula for calculating the "automeasure" changes; the mechanics of the process remain the same.

Autocorrelation has a number of uses. As we mentioned in Chapter 3, the autocorrelation peak can be used to determine the proper width of the grid for the search reduction technique of gridding. The value of the autocorrelation threshold can also be included in the vectors used in the technique of similarity, since similar areas really ought to have similar autocorrelations. Autocorrelation surfaces help to determine whether or not a given target area is suitable for matching. Most valuable, perhaps, is deciding whether or not a match is good, either for isolated matches or for region growing.

THE MATCH SUBROUTINE

Another basic part of correlation usage is the local strategy used to search for a matching point in an area thought to be promising. Most of our algorithms for determining whether or not an area is promising are based on whether the center of the area looks promising. Therefore, it makes sense, when considering the area in detail, to look first at points near the center and gradually work out toward the edges of the area. We have already observed that the correlation need not be calculated at every point of an area--calculating the correlation over a grid is adequate.

Based on these observations, the following local search algorithm was devised [Quam, 1971] to seek the highest correlation within a square area. The algorithm is implemented as a subroutine called MATCH, which takes four arguments. The first two are the coordinates of the center point of the area to be searched; when the routine returns, these variables contain the coordinates of the point found to have the highest correlation. The third argument gives the radius to which the search will be carried out; the fourth tells what value of correlation is to be the threshold for search termination.

As shown by Illustration B-3, the search starts at the center point of the candidate area, then spirals outward in the pattern indicated. At each point marked with a *, the correlation is calculated with the target area. The point having the highest correlation found so far is kept track of. Should the correlation exceed the preset threshold or the search radius be reached, the search stops spiraling and goes into hill-climbing mode at the point which had the highest correlation.

In hill-climbing mode, the algorithm examines the correlation at each of the eight points immediately surrounding the present point, and moves to the point which has the highest correlation. This loop is repeated until there is no higher point to move to, i.e. the summit of the hill has been reached.

The grid for the spiral is determined by a table within the routine. Originally, Quam set the table so that the algorithm used a grid spacing of 2 for the first loop, 3 for the next 3 loops, 4 for 2 loops, then 5. This author has implemented a version which uses a constant grid spacing for all loops, which is communicated by a global variable MGRID. This parameter is set by a routine which examines the autocorrelation peak, as explained in Chapter 3.

THE LMATCH SUBROUTINE

MATCH is a two-dimensional search strategy. When the area of interest has been confined to a line, however, we need a one-dimensional version, LMATCH. LMATCH has five arguments. The first four are the same as for MATCH, except that the center point is expressed as a real point lying on the given line. The fifth argument is the slope of the given line.

The search starts by calculating the correlation at the picture point closest to the given center point. It then moves n units up the line from the given starting point and calculates the correlation at the closest picture point, then repeats this n units down the line from the starting point, then 2n units up the line from the starting point, then 2n down, then 3n up, then 3n down, etc. Again, n is determined from the autocorrelation and communicated by MGRID.

Like MATCH, LMATCH keeps track of the best correlation found so far and exits from this "ping-pong" spiral when it reaches the radius or finds a correlation above the threshold. From the point having the best correlation, it goes into "inchworm" climbing mode, moving along the line in the uphill direction until it can't go up any more. Then it goes into the two-dimensional hill climb of MATCH, just in case the line was a little off and the matching point is not exactly on the line.

INTERPOLATION

It should be noted that all of the above techniques use correlation over areas centered on integer points in the picture. In practice, however, the proper match (in the sense of the candidate area which represents the same piece of the scene as the target area) for a given target will be an area centered on a point in Picture Y with non-integer co-ordinates. Since the only correlation values which are available are those at integer points, some form of interpolation is necessary whenever high precision is desired.

Therefore, the final operation on a match destined to be used for depth, camera model, or world model determination is an interpolation. We would like to fit a function of the form

$$\mathrm{EXP}(\, -\, (A*DI^2 + B*DJ + C*DI*DJ + D*DJ^2 + E*DJ + F)\,) \qquad \text{(B-e)}$$

to the correlation values C(X,Ix,Jx; Y,Iy+DI,Jy+DJ) for (DI,DJ) within some radius of the matching center points. To do this, we fit the polynomial $A*DI^2 + B*DI + C*DI*DJ + D*DJ^2 + E*DJ + F$ to the logarithm of the correlation values. Solving this function for a maximum gives the interpolated non-integer center point for the matching area in Picture Y.

When a model of the autocorrelation surface is desired, this same exponential fitting process is applied. Rather than being used to interpolate the autocorrelation, this exponential surface is used as an approximation to the autocorrelation peak. Examination of the coefficients of Equation B-e provide an easy way to determine the width of the peak, whether for calculation of the grid spacing or determination of the suitability of the area for matching.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1
1	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1
1	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1
1	1	3	3	3	5	5	5	5	5	5	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	5	5	5	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	5	5	5	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	7	7	7	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	7	7	7	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	7	7	7	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	5	5	5	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	5	5	5	5	5	5	3	3	3	1	1
1	1	3	3	3	5	5	5	5	5	5	5	5	5	3	3	3	1	1
1	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1
1	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1
1	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Linearly weighted window.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
1	1	2	2	2	4	4	4	4	4	4	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	4	4	4	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	4	4	4	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	8	8	8	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	8	8	8	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	8	8	8	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	4	4	4	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	4	4	4	4	4	4	2	2	2	1	1
1	1	2	2	2	4	4	4	4	4	4	4	4	4	2	2	2	1	1
1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Exponentially weighted window.

Illustration B-1. Windows of weights, such as might be used when minor distortions are present.

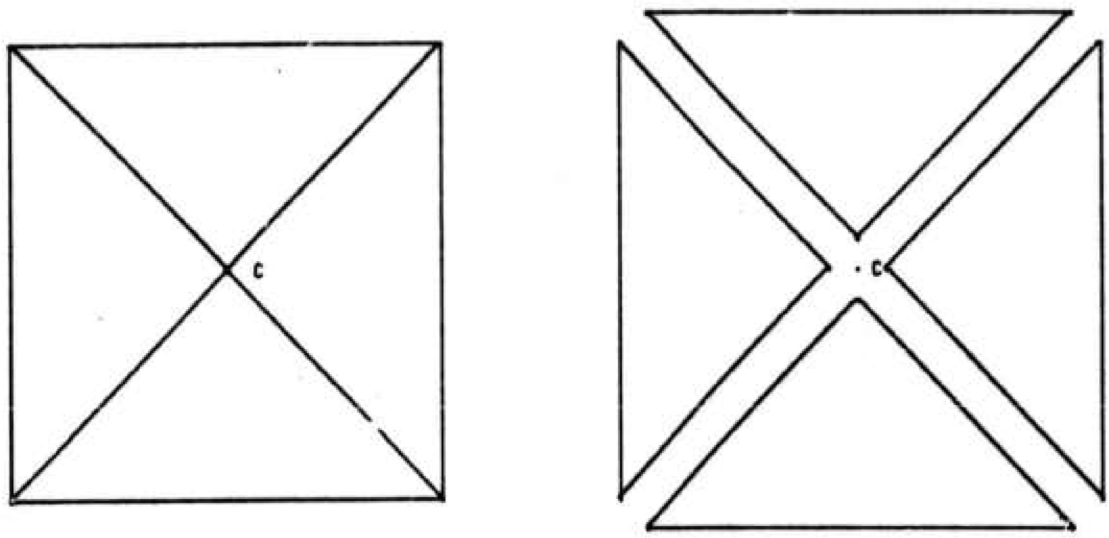


Illustration B-2. A sketch showing the manner in which a window could be distorted to determine an autocorrelation threshold over it. Pixels within the four areas spaced about the center point C as shown in the left drawing are correlated with pixels in the areas spaced about C as shown in the right drawing.

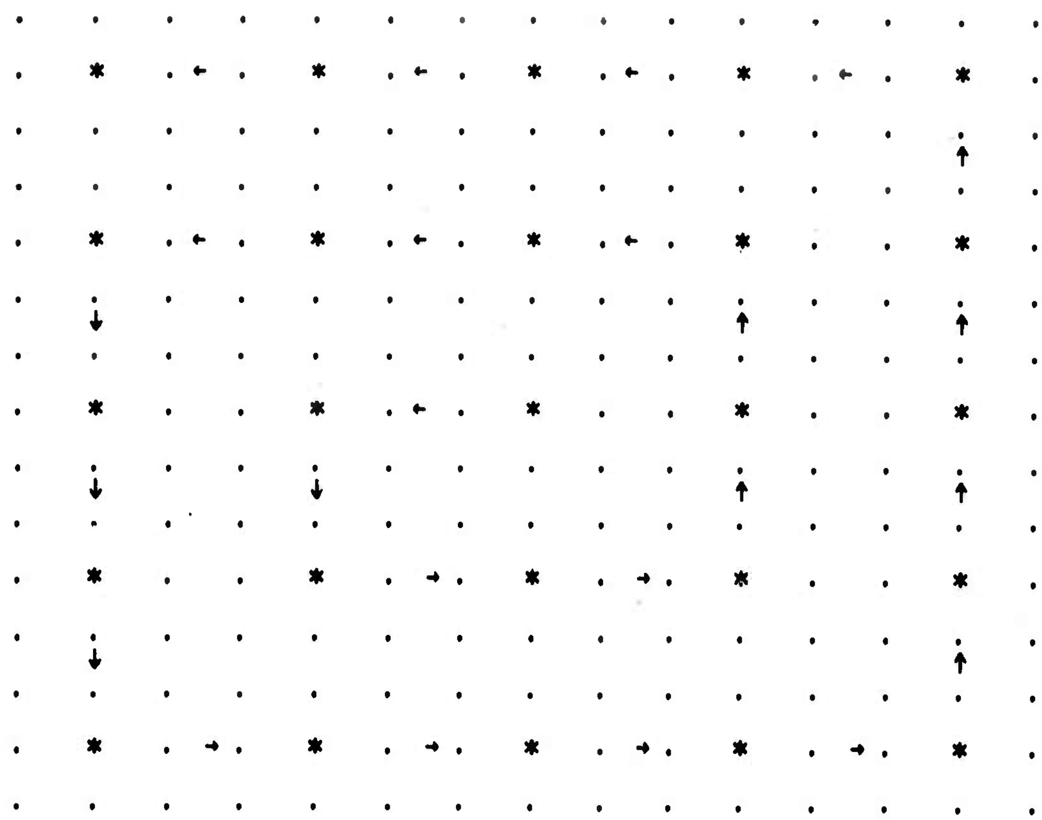


Illustration B-3. A representation of the search pattern for the subroutine MATCH. The algorithm begins at the center point and spirals outward following the arrows and calculating correlations at the points marked *. It stops spiralling when it finds a sufficiently high correlation or reaches the radius of the spiral.

Appendix C

CAMERA MODEL CALCULATIONS

For our purposes, a camera model consists of seven numbers which specify the principal distances of the two cameras and the position and orientation of the second camera with respect to the first. (The principal distance of a camera is the distance between its image plane and its principal point along its principal axis as shown in Illustration C-1). This appendix contains the mathematics used in deriving and utilizing camera models.

DERIVATION OF CAMERA MODEL EQUATIONS

We begin by arbitrarily placing a left-handed 3-dimensional co-ordinate system on the world in the following manner. The origin of this co-ordinate system is the principal point of the first camera. The principal axis of the camera becomes the z-axis of the world. The scale of the co-ordinate system is such that one unit equals the width of one pixel on the image plane. (See Illustration C-1)

Mathematically, the principal point has position (0,0,0); a point on the principal axis is represented by d*(0,0,1), and the image plane has the equation z=F1. The I- and J-axes of the first camera plane are parallel to the X- and Y-axes of the reference co-ordinate system, respectively, and in the plane z = F1, that is,

$$\vec{I} = (0,0,F1) + Ix*(1,0,0) \qquad \text{and} \qquad \vec{J} = (0,0,F1) + Jx*(0,1,0)$$

The principal point of the second camera is the point in 3-space described by the baseline distance D, which is the distance between the principal points of the two cameras, and by two angles, $\alpha 1$ and $\alpha 2$. When the first camera has been panned by $\alpha 1$ radians, then tilted by $\alpha 2$ radians, its principal axis will point down the baseline toward the principal point of the second camera. (See Illustration C-2)

Mathematically, panning is equivalent to a rotation about the Y-axis; tilting is equivalent to a rotation about the X-axis. The vector U is obtained by taking the vector (0,0,1), pre-multiplying it by the matrix $Rx(\alpha 2)$, representing a rotation of $\alpha 2$ degrees about the X-axis, pre-multiplying this result by the matrix $Ry(\alpha 1)$, representing a rotation of $\alpha 1$ about the Y-axis, and finally multiplying this quantity by the scalar D, i.e.

$$\vec{U} = D*(\ Ry(\alpha1)*(\ Rx(\alpha2)*(0,0,1)\ )\ )$$

$$= Ry(\alpha1) * Rx(\alpha2) * (0,0,D)$$

$$= \begin{bmatrix} COS(\alpha1) & 0 & SIN(\alpha1) \\ 0 & 1 & 0 \\ -SIN(\alpha1) & 0 & COS(\alpha1) \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & COS(\alpha2) & SIN(\alpha2) \\ 0 & -SIN(\alpha2) & COS(\alpha2) \end{bmatrix} * \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

where matrix multiplication is denoted by * and done in the usual fashion. Performing these multiplications, we have

$$\vec{U} = D*(\ SIN(\alpha1)*COS(\alpha2),\ SIN(\alpha2),\ COS(\alpha1)*COS(\alpha2)\ ) \tag{C-a}$$

The principal axis of the second camera is described by two more angles $\beta1$ and $\beta2$. When the first camera has been panned by $\beta1$ radians, then tilted by $\beta2$ radians, its axis parallels the axis of the second camera. (See Illustration C-3) A point on the principal axis is given by the position vector $\vec{U} + s*\hat{n}$, where s is the distance from the principal point $\vec{U}$, and $\hat{n}$ is a unit vector in the direction of the principal axis of the second camera.

Mathematically, $\hat{n}$ is expressed by pre-multiplying the vector (0,0,1) by the appropriate rotation matrices $Ry(\beta1)$ and $Rx(\beta2)$, i.e.

$$\hat{n} = Ry(\beta1)*(\ Rx(\beta2)*(0,0,1)\ )$$

$$= Ry(\beta1) * Rx(\beta2) * (0,0,1)$$

$$= \begin{bmatrix} COS(\beta1) & 0 & SIN(\beta1) \\ 0 & 1 & 0 \\ -SIN(\beta1) & 0 & COS(\beta1) \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & COS(\beta2) & SIN(\beta2) \\ 0 & -SIN(\beta2) & COS(\beta2) \end{bmatrix} * \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Performing these multiplications, we have

$$\hat{n} = (\ SIN(\beta1)*COS(\beta2),\ SIN(\beta2),\ COS(\beta1)*COS(\beta2)\ )$$

The image plane of the second camera is the plane perpendicular to the principal axis at distance F2 from the principal point. (See Illustration C-3) According to a standard analytic geometry textbook [Schwartz, 1960], the plane perpendicular to the vector $\vec{B}$ and passing through the point $\vec{P}$ has the equation

$$\vec{B} \cdot ((x,y,z) - \vec{P}) = 0 \quad .$$

Our image plane is defined to be the plane perpendicular to the principal axis $\hat{h}$ and passing through the point $\vec{U} + F2*\hat{h}$. Substituting these for $\vec{B}$ and $\vec{P}$, respectively, yields

$$\hat{h} \cdot ((x,y,z) - \vec{U} - F2*\hat{h}) = 0$$

The actual orientation of the second image plane is described by the angle $\beta 3$ through which the first image plane must roll (after having been panned and tilted to make the principal axes parallel) in order to make the internal co-ordinate axes of the first camera agree with those of the second camera. (See Illustration C-4) Let the I- and J-axes of the second camera plane be represented by the unit vectors $\hat{f}$ and $\hat{g}$, respectively.

The orientations of $\hat{f}$ and $\hat{g}$ depend on the pan and tilt angles $\beta 1$ and $\beta 2$, as well as the roll angle $\beta 3$. Mathematically, a roll is equivalent to a rotation about the Z-axis. Let $Ry(\beta 1)$ be the rotation matrix corresponding to panning by $\beta 1$, $Rx(\beta 2)$ be the rotation matrix corresponding to tilting by $\beta 2$, and $Rz(\beta 3)$ be the rotation matrix corresponding to rolling by $\beta 3$, i.e.

$$Ry(\beta 1) = \begin{bmatrix} COS(\beta 1) & 0 & SIN(\beta 1) \\ 0 & 1 & 0 \\ -SIN(\beta 1) & 0 & COS(\beta 1) \end{bmatrix} ,$$

$$Rx(\beta 2) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & COS(\beta 2) & SIN(\beta 2) \\ 0 & -SIN(\beta 2) & COS(\beta 2) \end{bmatrix} , \text{ and}$$

$$
Rz(\beta 3) = \begin{bmatrix} COS(\beta 3) & SIN(\beta 3) & 0 \\ -SIN(\beta 3) & COS(\beta 3) & 0 \\ 0 & 0 & 1 \end{bmatrix} .
$$

then we can express $\hat{f}$ and $\hat{g}$ as

$$
\hat{f} = Ry(\beta 1) * Rx(\beta 2) * Rz(\beta 3) * \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \text{ and}
$$

$$
\hat{g} = Ry(\beta 1) * Rx(\beta 2) * Rz(\beta 3) * \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} .
$$

Multiplying out these matrices in the usual fashion gives

$$
\begin{aligned}
\hat{f} = (\ & COS(\beta 1)*COS(\beta 3)+SIN(\beta 1)*SIN(\beta 2)*SIN(\beta 3), \\
& -COS(\beta 2)*SIN(\beta 3), \\
& COS(\beta 1)*SIN(\beta 2)*SIN(\beta 3)-SIN(\beta 1)*COS(\beta 3)\ ) \\
\hat{g} = (\ & COS(\beta 1)*SIN(\beta 3)-SIN(\beta 1)*SIN(\beta 2)*COS(\beta 3), \\
& COS(\beta 2)*COS(\beta 3), \\
& -SIN(\beta 1)*SIN(\beta 3)-COS(\beta 1)*SIN(\beta 2)*COS(\beta 3)\ )
\end{aligned}
$$

The I- and J-axes for the second camera radiate from the point $\bar{U} + F2*\hat{h}$, so we have

$$
\bar{I} = \bar{U} + F2*\hat{h} + Iy*\hat{f} \text{ and } \quad \bar{J} = \bar{U} + F2*\hat{h} + Jy*\hat{g} ,
$$

To derive a camera model, one takes a set of pairs of points found to be matches and searches in the space of F1, F2, α1, α2, β1, β2, and β3 for the values of these parameters which best accounts for these point-pairs.

Actual determination of the model is done by least-equares minimization of the measure of camera model error presented below. As in most least-squaree techniques, the number of point-pairs must be greater than N/2, where N is the number of parameters being sought, and should be independent points, i.e. no three co-linear in the image planes and no four representing co-planar points in 3-space. In practice, the number of reliable pairs available, p, should satisfy $p \geq 2N$, or in our case of $N=7$, $p \geq 14$. The program which derives the camera model sets an upper limit of 100 on the number of paire which can be ueed.

CAMERA MODEL ERROR MEASURE

There are many error measures possible. The one presented here is the average of the error in match for each of the point-pairs, calculated in the image plane. To calculate the error in match for each point-pair, we first use the first camera principal point to project point x of picture X into space as a ray, then use the second camera principal point for the hypothesized camera model to project this ray into the second image plane as a 2-dimensional line segment, and finally evaluate the distance in the second image plane between this line segment and the matching point y of picture Y.

Point x of Picture X is the point (Ix,Jx) in the plane of the first camera, which is the point $\vec{S}$ = (Ix,Jx,F1) in 3-space. The projection of this point into space is the ray from the principal point of the first camera, (0,0,0), through $\vec{S}$. In parameterized vector form, this ray is $r*\vec{S}$, $r>F1$.

Ueing the principal point of the second camera, this ray is projected into the image plane of the second camera. Perhaps the simplest way to derive this is to pick two arbitrary points on $r*\vec{S}$ and project them into the second camera image plane, then calculate the 2-dimensional line between them.

To facilitate this, first consider projecting an arbitrary point $\vec{Q}$ in 3-space into the plane P (in our case, the image plane of the second camera) perpendicular to the vector $\hat{n}$ (direction of the principal axis of the second camera) at the point $\vec{C} = \vec{U} + F2*\hat{n}$ (intersection of principal axis and second image plane) using the point $\vec{U}$ (principal point of the second camera) as the principal point of the projection. Clearly, the projected point lies at some distance t along the line from $\vec{Q}$ to $\vec{U}$, so can be described by the position vector $\vec{Q}' = \vec{U} + t*(\vec{Q}-\vec{U})$.

We would like to express $\vec{Q}'$ in terms of the vectors $\hat{f}$ and $\hat{g}$ which are ortho-normal and lie in the image plane P. That is, we would like to know I and J such that

$$\vec{U} + F2*\hat{n} + I*\hat{f} + J*\hat{g} = \vec{U} + t*(\vec{Q} - \vec{U}) \quad \text{or}$$

$$I*\hat{f} + J*\hat{g} = t*(\vec{Q} - \vec{U}) - F2*\hat{n} .$$

Dotting both sides of this vector equation by $\hat{f}$ gives

$$(I*\hat{f} + J*\hat{g}) \cdot \hat{f} = (t*(\vec{Q} - \vec{U}) - F2*\hat{h}) \cdot \hat{f} .$$

Expanding this, and using the fact that $\hat{f}$ is a unit vector and e perpendicular to both $\hat{h}$ and $\hat{g}$, we have

$$I = t*(\vec{Q} - \vec{U}) \cdot \hat{f} .$$

Had we dotted both sides of the equation by $\hat{g}$, we would have

$$J = t*(\vec{Q} - \vec{U}) \cdot \hat{g} .$$

Dotting both sides by $\hat{h}$ would give

$$t*(\vec{Q} - \vec{U}) \cdot \hat{h} - F2 = 0 \qquad \text{or}$$

$$t*(\vec{Q} - \vec{U}) \cdot \hat{h} = F2 \qquad \text{or}$$

$$t = F2/(\vec{Q} - \vec{U}) \cdot \hat{h} .$$

Substituting this expression for t into the expressions for I and J, we have

$$I = F2 * \frac{(\vec{Q} - \vec{U}) \cdot \hat{f}}{(\vec{Q} - \vec{U}) \cdot \hat{h}} \quad \text{and} \quad J = F2 * \frac{(\vec{Q} - \vec{U}) \cdot \hat{g}}{(\vec{Q} - \vec{U}) \cdot \hat{h}} .$$

Now we are ready to project two arbitrary points on the ray $r*\vec{S}$ into the plane P using the above equations. In the co-ordinates of the second image plane, the points $c0*\vec{S}$ and $c1*\vec{S}$ become the points (x0,y0) and (x1,y1), respectively, where

$$(x0,y0) = (F2 * \frac{(c0*\vec{S} - \vec{U}) \cdot \hat{f}}{(c0*\vec{S} - \vec{U}) \cdot \hat{h}} , F2 * \frac{(c0*\vec{S} - \vec{U}) \cdot \hat{g}}{(c0*\vec{S} - \vec{U}) \cdot \hat{h}}) \qquad \text{and}$$

$$(x1,y1) = (F2 * \frac{(c1*\vec{S} - \vec{U}) \cdot \hat{f}}{(c1*\vec{S} - \vec{U}) \cdot \hat{h}} , F2 * \frac{(c1*\vec{S} - \vec{U}) \cdot \hat{g}}{(c1*\vec{S} - \vec{U}) \cdot \hat{h}}) .$$

According to our analytic geometry text (Schwartz, 1960), the equation for the 2-dimensional line through these two points is

$$y - y1 = \frac{(y1 - y0)}{(x1 - x0)} * (x - x1) \qquad \text{or}$$

$$(y1 - y0) * x + (x0 - x1) * y + (y0*x1 - y1*x0) = 0 .$$

Evaluating (y1 - y0), we have

$$(y1 - y0) = F2 \ast \frac{(c1\ast\vec{S} - \vec{U}) \cdot \hat{g}}{(c1\ast\vec{S} - \vec{U}) \cdot \hat{h}} - F2 \ast \frac{(c0\ast\vec{S} - \vec{U}) \cdot \hat{g}}{(c0\ast\vec{S} - \vec{U}) \cdot \hat{h}}$$

$$= F2 \ast \frac{(c1\ast\vec{S}-\vec{U})\cdot\hat{g} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{h} - (c0\ast\vec{S}-\vec{U})\cdot\hat{g} \ast (c1\ast\vec{S}-\vec{U})\cdot\hat{h}}{(c1\ast\vec{S}-\vec{U})\cdot\hat{h} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{h}}$$

$$= F2 \ast \frac{(c1 - c0) \ast (\vec{S}\cdot\hat{h} \ast \vec{U}\cdot\hat{g} - \vec{S}\cdot\hat{g} \ast \vec{U}\cdot\hat{h})}{(c1\ast\vec{S}-\vec{U})\cdot\hat{h} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{h}} .$$

Similar manipulations give

$$(x0 - x1) = F2 \ast \frac{(c1 - c0) \ast (\vec{S}\cdot\hat{f} \ast \vec{U}\cdot\hat{h} - \vec{S}\cdot\hat{h} \ast \vec{U}\cdot\hat{f})}{(c1\ast\vec{S}-\vec{U})\cdot\hat{h} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{h}} .$$

Substituting into (x1*y0 - y1*x0), we have

$$(x1\ast y0 - y1\ast x0) = F2\ast\frac{(c1\ast\vec{S} - \vec{U})\cdot\hat{f}}{(c1\ast\vec{S} - \vec{U})\cdot\hat{h}} \ast F2\ast\frac{(c0\ast\vec{S} - \vec{U})\cdot\hat{g}}{(c0\ast\vec{S} - \vec{U})\cdot\hat{h}}$$

$$- F2\ast\frac{(c1\ast\vec{S} - \vec{U})\cdot\hat{g}}{(c1\ast\vec{S} - \vec{U})\cdot\hat{h}} \ast F2\ast\frac{(c0\ast\vec{S} - \vec{U})\cdot\hat{f}}{(c0\ast\vec{S} - \vec{U})\cdot\hat{h}}$$

$$= F2^2\ast\frac{(c1\ast\vec{S}-\vec{U})\cdot\hat{f} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{g} - (c1\ast\vec{S}-\vec{U})\cdot\hat{g} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{f}}{(c1\ast\vec{S} - \vec{U})\cdot\hat{h} \ast (c0\ast\vec{S} - \vec{U})\cdot\hat{h}}$$

$$= F2^2 \ast \frac{(c1 - c0) \ast (\vec{S}\cdot\hat{g} \ast \vec{U}\cdot\hat{f} - \vec{S}\cdot\hat{f} \ast \vec{U}\cdot\hat{g})}{(c1\ast\vec{S} - \vec{U})\cdot\hat{h} \ast (c0\ast\vec{S} - \vec{U})\cdot\hat{h}} .$$

Now, substituting these terms into the equation for the line gives

$$F2 \ast \frac{(c1 - c0) \ast (\vec{S}\cdot\hat{h} \ast \vec{U}\cdot\hat{g} - \vec{S}\cdot\hat{g} \ast \vec{U}\cdot\hat{h})}{(c1\ast\vec{S}-\vec{U})\cdot\hat{h} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{h}} \ast x +$$

$$F2 \ast \frac{(c1 - c0) \ast (\vec{S}\cdot\hat{f} \ast \vec{U}\cdot\hat{h} - \vec{S}\cdot\hat{h} \ast \vec{U}\cdot\hat{f})}{(c1\ast\vec{S}-\vec{U})\cdot\hat{h} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{h}} \ast y +$$

$$F2^2 \ast \frac{(c1 - c0) \ast (\vec{S}\cdot\hat{g} \ast \vec{U}\cdot\hat{f} - \vec{S}\cdot\hat{f} \ast \vec{U}\cdot\hat{g})}{(c1\ast\vec{S}-\vec{U})\cdot\hat{h} \ast (c0\ast\vec{S}-\vec{U})\cdot\hat{h}} = 0 .$$

Factoring out common terms and dividing by them gives

$$(\vec{S}\cdot\hat{h} * \vec{U}\cdot\hat{g} - \vec{S}\cdot\hat{g} * \vec{U}\cdot\hat{h}) * x \ +$$

$$(\vec{S}\cdot\hat{i} * \vec{U}\cdot\hat{h} - \vec{S}\cdot\hat{h} * \vec{U}\cdot\hat{i}) * y \ +$$

$$(\vec{S}\cdot\hat{g} * \vec{U}\cdot\hat{i} - \vec{S}\cdot\hat{i} * \vec{U}\cdot\hat{g}) * F2 = 0 \quad . \qquad \text{(C-b)}$$

the desired line segment in the second image.

The error for that point-pair is the square of the minimum distance between this line segment which corresponds to the point x and the point y which matches the point x. (See Illustration C-5)

DEPTH RANGING

Once one has a camera model, it is relatively trivial to find the distance from either of the cameras to an object in 3-space represented by a point-pair. One has the points (Ix,Jx) and (Iy,Jy). The ray from the principal point of the first camera through (Ix,Jx) is given by the vector r*(Ix,Jx,F1). The ray from the principal point of the second camera through (Iy,Jy) is given by

$$\vec{U} + s *(F2*\hat{h} + Iy*\hat{i} + Jy*\hat{g}) \qquad \text{(C-c)}$$

Due to minor errors in measurements of camera model parameters or in interpolation of the matching center point, these two rays may not intersect. Using the camera model, we can correct for this. We first back-project the point x into the second image plane, giving us the line of Equation C-b. Now, instead of the point (Iy,Jy), we decree the point (Iy',Jy') which is on this line and which is the shortest distance away from (Iy,Jy) to be the true matching point. This gives us the ray

$$\vec{U} + s *(F2*\hat{h} + Iy'*\hat{i} + Jy'*\hat{g}) \qquad \text{(C-d)}$$

in lieu of Equation C-c.

To simplify the notation in the following derivation, let

$$\vec{P} = (Ix,Jx,F1)$$

$$\vec{Q} = F2*\hat{h} + Iy'*\hat{i} + Jy'*\hat{g} \quad .$$

We know that the intersection of $r * \vec{P}$ and $\vec{U} + s * \vec{Q}$ exists; that is the definition of (Iy',Jy'). Therefore, we need only solve for the r and s such that $r * \vec{P} = \vec{U} + s * \vec{Q}$. The two necessary constraints are obtained by dotting both sides of this equation by $\vec{P}$ or by $\vec{Q}$, ie.

$$(r * \vec{P}) \cdot \vec{P} = (\vec{U} + s * \vec{Q}) \cdot \vec{P} \qquad \text{and}$$

$$(r * \vec{P}) \cdot \vec{Q} = (\vec{U} + s * \vec{Q}) \cdot \vec{Q} \qquad .$$

These are equivalent to

$$r * \vec{P}\cdot\vec{P} = \vec{U}\cdot\vec{P} + s * \vec{Q}\cdot\vec{P} \qquad \text{and}$$

$$r * \vec{P}\cdot\vec{Q} = \vec{U}\cdot\vec{Q} + s * \vec{Q}\cdot\vec{Q} \qquad .$$

Solving this equation for s gives the distance of the 3-dimensional point from the second camera

$$s = \frac{\vec{U}\cdot\vec{Q} * \vec{P}\cdot\vec{P} - \vec{U}\cdot\vec{P} * \vec{P}\cdot\vec{Q}}{\vec{Q}\cdot\vec{P} * \vec{P}\cdot\vec{Q} - \vec{P}\cdot\vec{P} * \vec{Q}\cdot\vec{Q}} ,$$

while solving the above system for r gives the distance from the first camera

$$r = \frac{\vec{U}\cdot\vec{Q} * \vec{Q}\cdot\vec{P} - \vec{U}\cdot\vec{P} * \vec{Q}\cdot\vec{Q}}{\vec{Q}\cdot\vec{P} * \vec{P}\cdot\vec{Q} - \vec{P}\cdot\vec{P} * \vec{Q}\cdot\vec{Q}} .$$

DERIVATION OF TWO MATCHING LINES

With a camera model, it is possible to place two lines, one in each picture, into correspondence. To see this, consider the two principal points of the cameras, (0,0,0) and $\vec{U}$.

These two points, plus any third point, determine a plane in 3-space. If we call the third point $\vec{S}$, then this plane has as its normal the vector $\vec{S} \times \vec{U}$ and goes through the point (0,0,0). Our analytic geometry text tells us that the equation of a plane with normal $\vec{N}$ through the point $\vec{P}$ is

$$\vec{N} \cdot ((x,y,z) - \vec{P}) = 0$$

Therefore the equation of the plane determined by (0,0,0), $\vec{U}$, and $\vec{S}$ has the equation

$$(\vec{S} \times \vec{U}) \cdot (x,y,z) = 0 \qquad \text{(C-e)}$$

Except in a few degenerate cases, this plane intersects both of the camera image planes. The intersection of this plane with the second image plane in terms of that plane's coordinate system is given in Equation C-b; the Intersection with the first image plane z = F1 is

$$(\vec{S} \times \vec{U}) \cdot (x,y,F1) = 0 \qquad \text{(C-f)}$$

Consider also the intersection of the plane of Equation C-e with the scene. All of the points of this curve lie on the plane of Equation C-e, obviously; therefore all of the projections of these points onto the second image plane lie on the line of Equation C-b and all of the projections onto the first image plane lie on the line of Equation C-f. Thus all of the points on one line map to points on the other line.

Clearly, $\vec{S}$ can be almost any point. The most convenient such point is usually (Ix,Jx,F1), the center point of the target area.

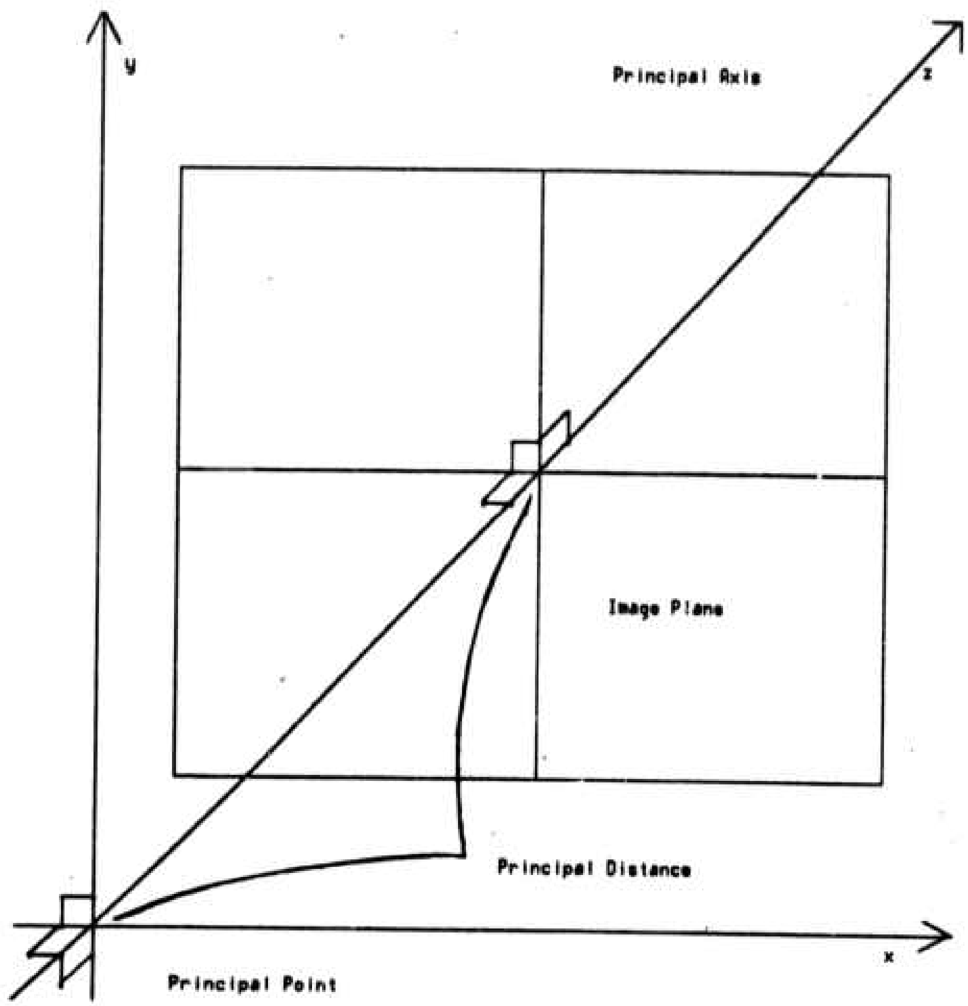


Illustration C-1. One of our simplified cameras in the standard position and orientation.

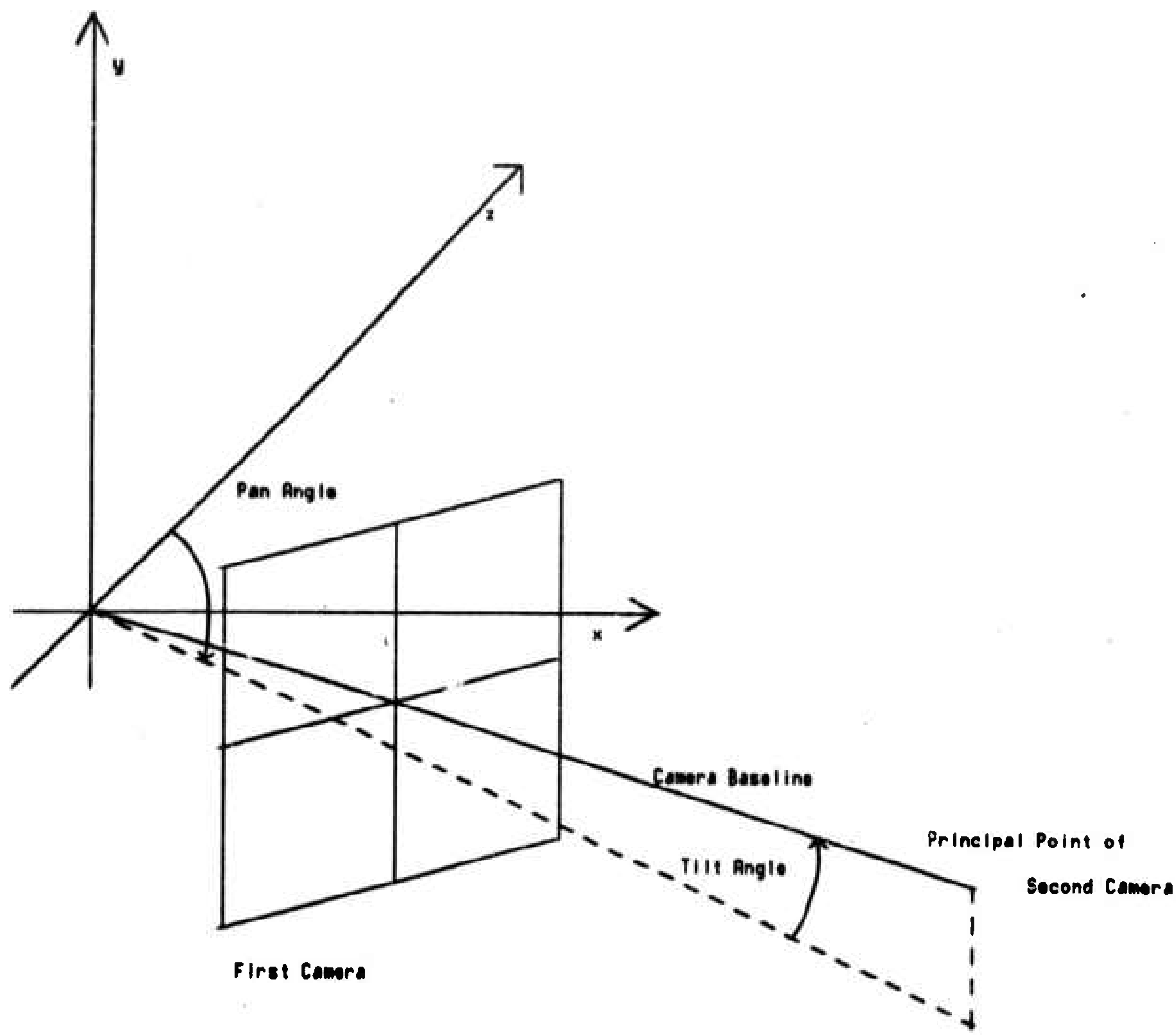


Illustration C-2. The first camera, panned and tilted to point to the principal point of the second camera.

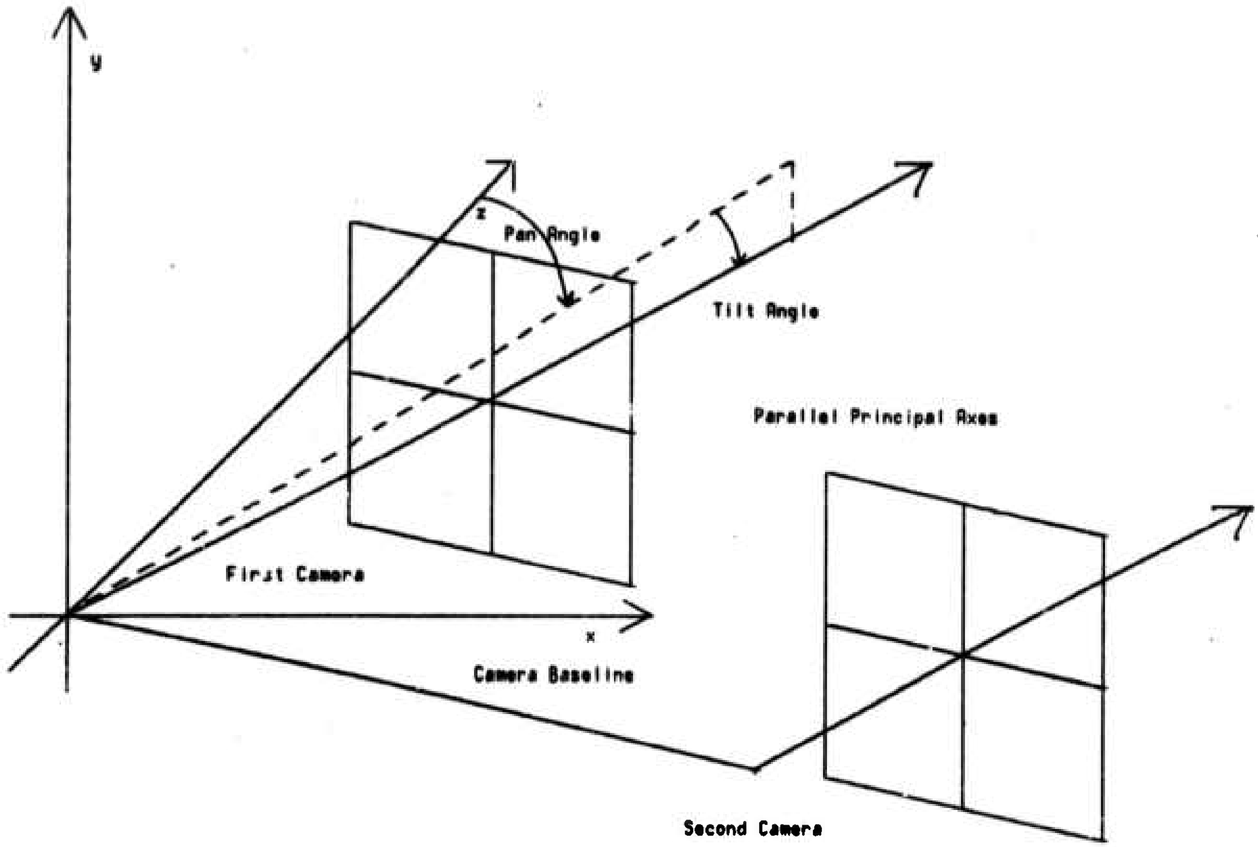


Illustration C-3. The first camera panned and tilted so its principal axis parallels the principal axis of the second camera.

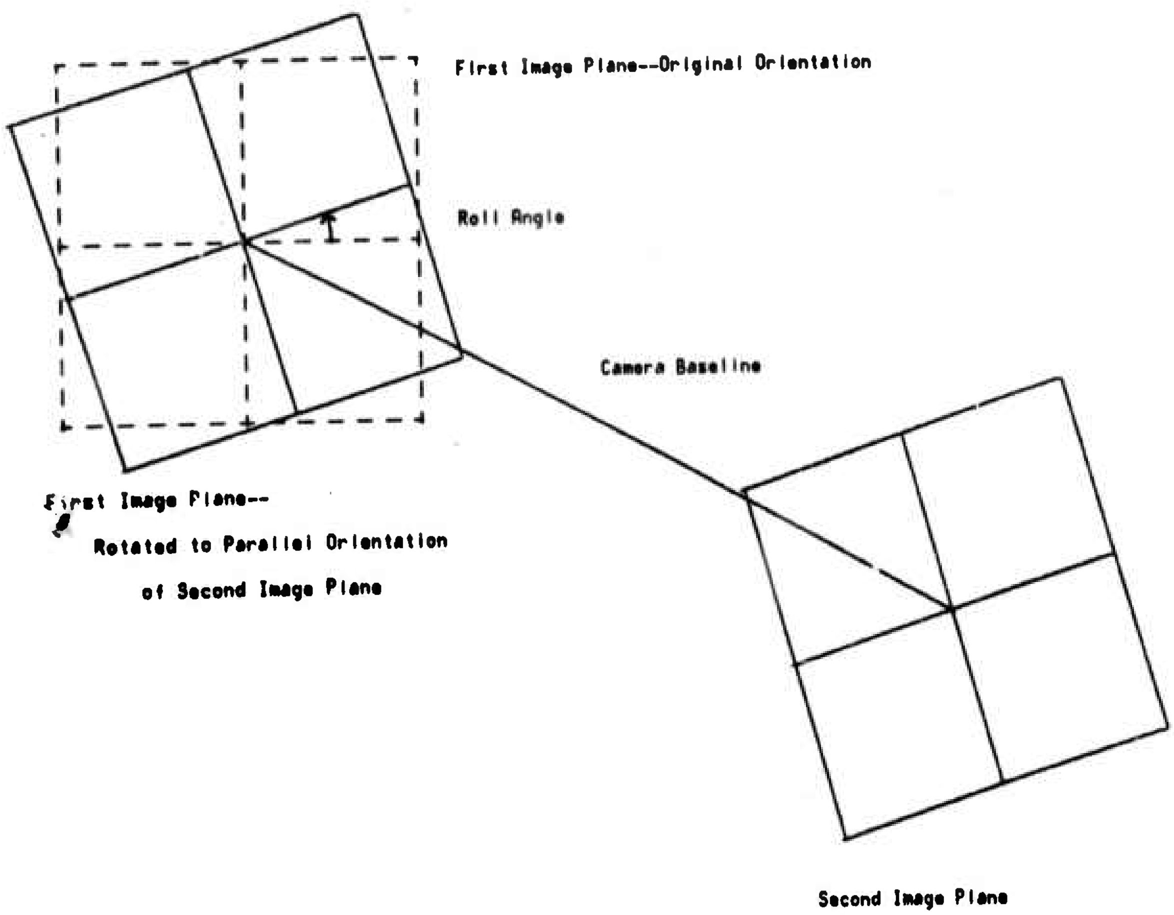


Illustration C-4. The first camera image plane rolled to parallel the second camera image plane.

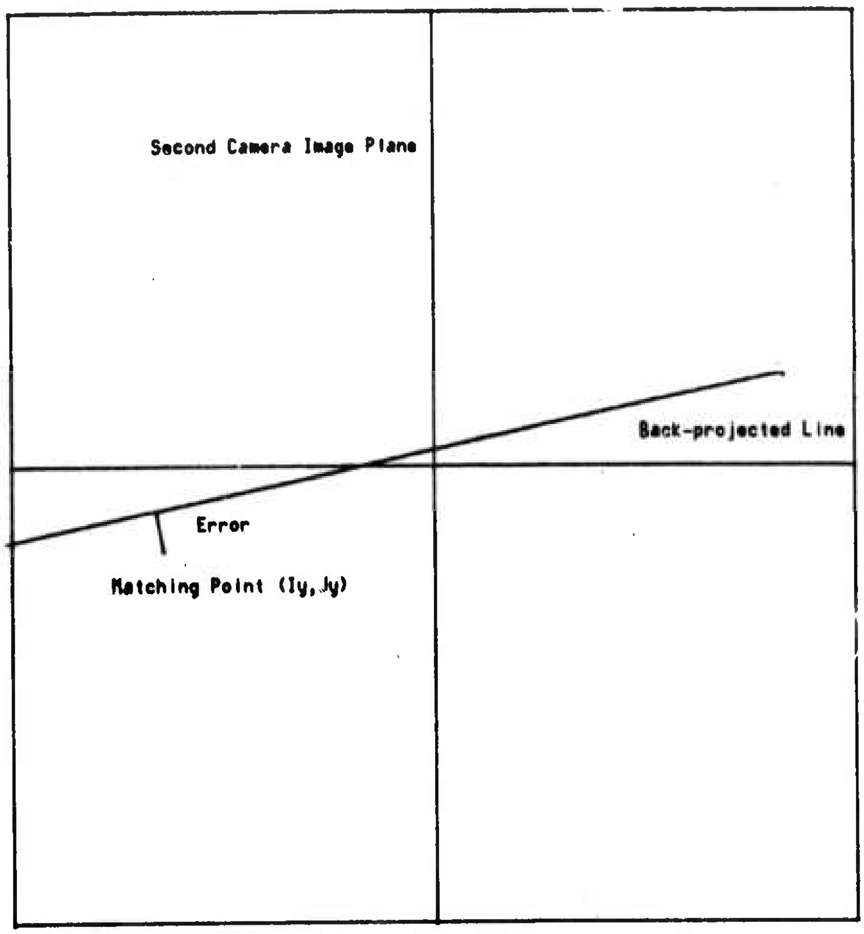


Illustration C-5. The error for a point pair (Ix,Jx) ↔ (Iy,Jy) is the distance from (Iy,Jy) to the line in the second image which corresponds to (Ix,Jx).

Appendix D

DISTORTION

Intuitively, if the parts of the two pictures which represent a given object differ in anything but position, then the object has been distorted from one view to the other. For our purposes, if, for displacements (di,dj) within some window and corresponding points (Ix,Jx) and (Iy,Jy) in the two images, the point (Ix+di,Jx+dj) does not correspond to the point (Iy+di,Jy+dj), there is distortion over that window.

MATHEMATICAL DESCRIPTION

To express this mathematically, we start with two points in 3-space, $\vec{R}$ and $\vec{S}$. According to the calculations in Appendix C, these points project to

$$\vec{R}1 = (R1x,\ R1y) = \left(\frac{\vec{R}\cdot\hat{i}}{\vec{R}\cdot\hat{k}} * F1 ,\ \frac{\vec{R}\cdot\hat{j}}{\vec{R}\cdot\hat{k}} * F1 \right) \qquad \text{and}$$

$$\vec{S}1 = (S1x,\ S1y) = \left(\frac{\vec{S}\cdot\hat{i}}{\vec{S}\cdot\hat{k}} * F1 ,\ \frac{\vec{S}\cdot\hat{j}}{\vec{S}\cdot\hat{k}} * F1 \right)$$

in the first image plane and

$$\vec{R}2 = (R2x,\ R2y) = \left(\frac{(\vec{R}-\vec{U})\cdot\hat{f}}{(\vec{R}-\vec{U})\cdot\hat{h}} * F2 ,\ \frac{(\vec{R}-\vec{U})\cdot\hat{g}}{(\vec{R}-\vec{U})\cdot\hat{h}} * F2 \right) \qquad \text{and}$$

$$\vec{S}2 = (S2x,\ S2y) = \left(\frac{(\vec{S}-\vec{U})\cdot\hat{f}}{(\vec{S}-\vec{U})\cdot\hat{h}} * F2 ,\ \frac{(\vec{S}-\vec{U})\cdot\hat{g}}{(\vec{S}-\vec{U})\cdot\hat{h}} * F2 \right)$$

in the second image plane.

Suppose we let $\vec{S}$ be the reference point, that is, we set (S1x,S1y) = (Ix,Jx) and (S2x,S2y) = (Iy,Jy). Also, let (R1x-S1x,R1y-S1y) be the (di,dj) of our intuitive definition. There is distortion if the point which corresponds to (Ix,Jx) + (di,dj) is not (Iy,Jy) + (di,dj), that is

$$(R2x,\ R2y) \neq (S2x,\ S2y) + (R1x-S1x,\ R1y-S1y) \qquad \text{or}$$

$$(S2x, S2y) - (R2x, R2y) + (R1x-S1x, R1y-S1y) = 0 \quad \text{or}$$

$$(R1x-S1x-R2x+S2x, R1y-S1y-R2y+S2y) = 0$$

We define this last vector to be $\vec{D}$, the distortion vector.

For non-trivial camera models and windows larger than a single point, it is unlikely that this vector will be exactly zero for all of the (di,dj) within the window. Consequently, there will almost always be distortion in a continuous image.

However, we are dealing, not with continuous images, but with images which are represented by discrete arrays. When, in such an array, the desired image point falls between elements of the image array, there are two things which can be done. One can approximate the desired pixel by interpolating the neighboring array elements, or one can simply use the array element which is closest to the desired point. In correlating, the latter is the more common practice.

The vector $\vec{D}1 = \vec{R}1 - \vec{S}1$ will ordinarily be such that if its tail is placed on an integer point of the array, its head will also fall onto an integer point. If the vector $\vec{D}2 = \vec{R}2 - \vec{S}2$ is placed with its tail on the same integer point as $\vec{D}1$, its head will probably not fall on the head of $\vec{D}1$. However, if the head of $\vec{D}2$ falls within 1/2 pixel in each co-ordinate of the head of $\vec{D}1$, we can not really tell the difference in position. Thus, for a discrete image, we can say that there is no distortion if, for all (di,dj) within the window, the x- and y-components of $\vec{D}$ are both less than 1/2 pixel.

LIMITING DISTORTION

Distortion is algebraically a very complicated quantity, for it depends on thirteen parameters--the pan and tilt angles which describe the direction to the second camera, the pan, tilt, and roll angles which describe the orientation of the second camera, the two focal lengths, and three parameters each to describe the relative 3-space points R and S. Graphing the distortion as a function of all 13 of these parameters is obviously not feasible; the graph is impossible to represent physically and excessively large to tabulate.

If one holds all but one of the parameters constant, one can use the limitation on the components of the distortion vector to solve for limits on the last parameter which will guarantee that the distortion is small. This is possible analytically (see [Fischler, 1971] for a treatment of change of focal length and for second camera roll angle), but is usually rather messy, hence not very illuminating. To give a feel for the results for particular parameters, Illustration D-1 tabulates some of the distortions for the camera model of the barn pictures with different object positions and orientations.

The barn camera model is almost the standard side-by-side stereo model. The second camera is placed at 81 degrees of pan from the first camera and .6 degrees of tilt, that is, to the right of the first camera, slightly forward of its position, and a little bit higher. Its pointing data is -3.2 degrees of pan, -1.3 degrees of tilt, and -1.4 degrees of roll, that is, it is pointed slightly to the left (back toward the first camera), down a little, with a minor clockwise roll.

The first group of data tabulates the distortion for two points in the first image plane (-50,10), which is on the corner of the barn door, and (-55,5), which is (-5,-5) pixels away. The depths to the corresponding 3-space points are kept equal, that is, the 3-space points are both on a plane perpendicular to the first camera's principal axis. As this depth increases from one meter to 100 meters, we observe the resulting changes in the distortion.

The second group of data uses a different pair of points in the first image, (10,10) and (17,17)--the point on the skyline where the trees show somewhat of a notch and a point (7,7) pixels away. For a depth of 10 meters at (10,10), we vary the depth at (17,17) on either side of 10 meters and observe the results.

In the third group of data, we have used the same first point (10,10) and varied the vector to the second point, in effect examining the effect of varying the window radius from 1 to 10 pixels. For each pair of points, we have determined (to two decimal places) the depth at which the two 3-space points would have to lie (both at the same depth) in order to produce distortion of just less than half of a pixel.

It is hoped that this table will give some feel for the relation between depth, window size, object orientation, and distortion. Those wishing to draw specific conclusions about the allowable window size, etc., for their own data are advised to program the mathematics of the last section and produce similar tables for their camera model, since the distortion vectors will change considerably with changes in the camera model parameters.

Under the definition of the last section, depth discontinuities are distortions. However, such discontinuities are effectively translatione, which our algorithms can handle once they are located, so we will exclude depth discontinuities from the following discussions.

SMALL DISTORTIONS

For known small rotations and scale factor changes it is possible to choose the correlation window to be distortion-free [Fischler, 1971]. This is done by calculating at what radius the global distortion causes pixels of matching windows to get one pixel out of registration, yielding local distortion. Any window which would fit into a square of this radius will be distortion-free, at least from this source of distortion.

For other minor distortions, weighting the correlation window as shown in Illustration B-1 may also help. (See Appendix B for an explanation of weighted correlation.) Essentially, this says that we are most interested in having the center of the correlation window match up, and, while it would be nice to have the outer parts match up, it should not greatly effect the correlation if they do not.

GROSS DISTORTIONS

But what about large rotations and scale factor changes? Large distortions will cause matching to fail, since it causes the matching process to compare points which do not correspond. When enough points do not correspond, the correlation will fall below the confidence level, and the areas will fail to match. This necessitated our restrictions on the kind of pictures we can handle.

However, some of these restrictions can be lifted. The main technique for this consists of figuring out what caused the problem and compensating for it. Let us consider some of the causes of large distortions and see what can be done about them.

Global Rotations and Scale Factor Changes

Global rotations and scale factor changes--those affecting the whole picture--are caused by a relative roll of one camera about its focal axis and by differences in the focal lengths of the cameras, respectively. Pairs of pictures having these distortions are somewhat rare. The human prejudice for order usually results in multiple photographs of a scene being taken with identical cameras and lenses, and with both cameras held upright.

There exists the case in which the pictures were taken by a machine, such as an independent roving vehicle. However, a reasonable design constraint on such a machine is for it to monitor its orientation with respect to the world, and report how much roll is present if it must change angles. One would also expect to know if the focal length of one camera differed from that of the other. Given this data, it is possible to decalibrate the pictures, that is, put them into the same orientation and scale [Quam, 1971].

In the rare case in which gross rotations or scale factor changes are present but of unknown magnitude, it is still possible to get rid of them. All that is required is to determine the rotation and scale factor change.

If the locations of enough pairs of points were known, the global rotation and scale factor change could be computed by least squares techniques as a part of the camera model (See Appendix C). This requires collecting several pairs of corresponding points. Since we have assumed that distortions exist, we cannot use matching techniques, which depend on low distortion, to find these point pairs.

One possible method for discovering these correspondences is to extend the correlation technique. Instead of merely searching among all possible translations of the window, Iy = Ix + CI and Jy = Jx + CJ, we could also searches among all possible rotations and scale factor changes.

$$IY = S*(COS(\beta)*IX + SIN(\beta)*JX) + CI \quad \text{and}$$

$$JY = S*(-SIN(\beta)*IX + COS(\beta)*JX) + CJ$$

These new dimensions, S and β, will have to be quantized in order to make the searches finite. The window size used for the correlation will determine the maximum quantization possible without having to worry about distortion.

This search in 4 variables will be very long and slow; some method of shortening it is almost manditory. The technique of reduction will still work if the size of the window can be reduced along with the picture. Gridding will also still work for the translation part of the search, and is inherent in the quantization of the rotation angle and scale factor. Similarity will work only if the properties put into the vectors are invariant under rotation and scale factor change. Camera model searches are not applicable, since we have no camera model. (If we did, we would know the the relative rotation and focal lengths, and wouldn't be looking for them the hard way.)

Another method which will give the rotations and scale factor changes directly was suggested by Lynn Quam. It calls for locating some object or area lying entirely in both pictures and finding its boundary. This could be accomplished by flat region growing (see Chapter 5) for an area of low variance, by conventional edge techniques [Hueckel, 1972], or by more sophisticated region growing techniques [Yakimovsky, 1973]. The boundary is then tabulated as distance from the center of mass of the area vs. angle from some reference direction. It is now possible to correlate the resulting function tables to find the optimal displacement (i.e. angle) which aligns them. Once the rotational alignment is determined, the tabulated distances at corresponding points on the boundaries can be used to derive the scale factor change, as could the ratio of the perimeters.

To the knowledge of this author, these techniques have not been implemented. Since totally unknown camera roll and focal length change tend to be the exception, rather than the rule, this author leaves the solution to someone who has the problem.

Local Scale Factor Changes

Local scale factor changes occur because the object is closer to one camera than to the other. This is particularly noticeable in forward motion stereo, as might be taken by a vehicle rolling along some path. If the approximate local scale factor is known, it can be taken into account, and a correlation function which does mapping instead of matching can be employed to determine the area correspondence. (See Appendix B for descriptions of matching and mapping correlation.)

The idea of finding boundaries and thus determining the relative scale factor change is still feasible; however, it requires knowing where the object is in both pictures. Since this might well be the information we are trying to determine, this approach is usually not practical.

A second technique recently implemented by Quam uses a camera model Given a camera model and a pair of points, it is computationally rather simple to determine the distance from each camera to the point in 3-space corresponding to those two points (See Appendix C). For each proposed mapping, these distances are calculated using the center points of the proposed corresponding areas. From these distances and the focal lengths, one calculates the effective difference in scale between the two areas so that mapping tables can be set and the correlation evaluated.

When the object lies at the same distance from both cameras, but with a face at a large angle to the camera baseline, scale distortion occurs primarily in one dimension in the image--the dimension most nearly parallel to the camera baseline. For instance, in the barn pictures the barn face is distorted from one view to the other, but the distortion is primarily horizontal--the direction of the camera baseline. This suggests making the correlation window more narrow in the direction of the camera baseline in order to reduce distortion to allow matching to take place.

Most other distortions cause the two images to contain different projections of the object. In general, it is unlikely that more than a small part of an object so distorted can be matched. Mapping is possible, of course--IF one has a 3-D model of the object, knows the original 3-space position and orientation of the object and how it changed, and has a reliable camera model. However, if one already knows that much about the scene, there is little point in doing matching, or any other vision work.

For a given pair of points (Ir,Jr) and (Is,Js), examine the distortion vector (Dx,Dy) as a function of depth, r-depth = s-depth, i.e. all of the 3-space points are in a plane perpendicular to the first camera's principal axis.

(Ir,Jr)	r-depth	(Is,Js)	s-depth	Dx	Dy
-50 10	1.000	-55 5	1.000	-.127	-.565
-50 10	1.500	-55 5	1.500	-.113	-.307
-50 10	2.000	-55 5	2.000	-.099	-.186
-50 10	5.000	-55 5	5.000	-.065	.017
-50 10	10.000	-55 5	10.000	-.051	.082
-50 10	15.000	-55 5	15.000	-.046	.103
-50 10	20.000	-55 5	20.000	-.044	.113
-50 10	50.000	-55 5	50.000	-.040	.132
-50 10	100.000	-55 5	100.000	-.038	.138

For a given pair of points and r-depth, examine the distortion vector as the s-depth varies.

(Ir,Jr)	r-depth	(Is,Js)	s-depth	Dx	Dy
10 10	10.000	17 17	10.060	.515	-.028
10 10	10.000	17 17	10.050	.474	-.027
10 10	10.000	17 17	10.000	.272	-.023
10 10	10.000	17 17	9.900	-.138	-.015
10 10	10.000	17 17	9.820	-.472	-.009
10 10	10.000	17 17	9.810	-.514	-.008

For given (Ir,Jr) and r-depth = s-depth, find approximate depth at which the maximum distortion of .5 occurs for a variety of (Is,Js).

(Ir,Jr)	r-depth	(Is,Js)	s-depth	Dx	Dy
10 10	.380	11 11	.380	-.024	.487
10 10	.610	12 12	.610	.105	.492
10 10	.820	13 13	.820	.176	.499
10 10	1.030	14 14	1.030	.231	.496
10 10	1.220	15 15	1.220	.281	.499
10 10	2.240	20 20	2.240	.500	.448

Illustration D-1. A table of the distortion vectors in the barn pictures for different object positions and orientations. The depths given are in meters and repreeent the z-coordinate of that particular 3-space point.

BIBLIOGRAPHY

Barnea, Daniel I. and Harvey F. Silverman [1972], "A Class of Algorithms for Fast Digital Image Registration", *IEEE Transactions on Computers*, Vol. C-21, No. 2, February, 1972, pp. 179-186.

Bouchard, Harry and Francis H. Moffitt [1965], "Photogrammetry", *Surveying*, pp. 482-519, International Textbook Co., Scranton, Pa.

Cooley, J.W., P.A.W Lewis, and P.D. Welch [1967], "Applications of Fast Fourier Transform to Computation of Fourier Integrals, Fourier Series, and Convolution Integrals", *IEEE Transactions on Audio and Electroacoustics*, Vol. AU-15, No. 2, June, 1967, pp. 79-84.

Falk, Gilbert [1969], "Some Implications of Planarity for Machine Perception", Stanford Artificial Intelligence Project Memo No. 107.

Feldman, J. A., G. M. Feldman, G. Falk, G. Grape, J. Pearlman, I. Sobel, J. M. Tenenbaum [1969], "The Stanford Hand-Eye Project", *Proceedings of the International Joint Conference on Artificial Intelligence*, May, 1969, pp. 521-526.

Fischler, Martin A. [1971], "Aspects of the Detection of Scene Congruence", *Proceedings of the Second Annual International Joint Conference on Artificial Intelligence*, September, 1971, pp. 88-100.

Fischler, M. A., and R. A. Elschlager [1971], "The Representation and Matching of Pictorial Structures", Lockheed Missiles and Space Company, LMSC-D243781, September, 1971.

Freund, John E. [1962], *Mathematical Statistics*, Prentice-Hall, Inc., Englewood Cliffs, N.J.

Guzman, Adolfo [1968], "Computer Recognition of Three-Dimensional Objects in a Visual Scene", MIT Project MAC MAC-TR-59.

Hueckel, Manfred H. [1969], "An Operator Which Locates Edges in Digitized Pictures", Stanford Artificial Intelligence Project Memo No. 105.

Julesz, B. [1961], "Binocular Depth Perception and Pattern Recognition", *Proceedings of the Fourth London Symposium on Information Theory*, pp. 212-224.

Julesz, B. [1963], "Towards the Automation of Binocular Depth Perception", *Proceedings of IFIP Congress, 1962*, North Holland Publishing Co. Amsterdam, pp. 439-443.

Kelly, Michael D. [1970], "Visual Identification of People by Computer", Stanford Artificial Intelligence Project Memo No. 130.

Lathi, B. P. [1968], *An Introduction to Random Signals and Communication Theory*, International Textbook Company, Scranton, Pa.

Levine, M. D., D. A. O'Handley, and G. M. Yagi [1973], "Computer Determination of Depth Maps", Jet Propulsion Laboratory, Pasadena, Ca.

Nilsson, Nils J. [1971], *Problem Solving Methods in Artificial Intelligence*, McGraw-Hill, San Francisco, Ca.

Quam, Lynn H. [1971], "Computer Comparison of Pictures", Stanford Artificial Intelligence Project Memo No. 144.

Quam, Lynn H., Sidney Liebes, Jr., Robert B. Tucker, Marsha Jo Hannah, and Botond G. Eross [1972], "Computer Interactive Picture Processing", Stanford Artificial Intelligence Project Memo No. 166.

Roberts, L.G. [1963], "Machine Perception of 3-Dimensional Solids", MIT Lincoln Laboratories Technical Report No. 315.

Rosenfeld, Azriel [1969a], *Picture Processing by Computer*, Academic Press, New York, N. Y. .

Rosenfeld, Azriel [1969b], "Picture Processing by Computer", *Computing Surveys*, Vol. 1, No. 3, September, 1969, pp. 147-174.

Rosenfeld, Azriel [1973], "Progress in Picture Processing: 1969-71", *Computing Surveys*, Vol. 5, No. 2, June, 1973, pp. 81-108.

Schwartz, Abraham [1960], *Analytic Geometry and Calculus*, Holt, Rinehart and Winston, Inc., New York, N.Y.

Singleton, Richard C. [1967], "On Computing the Fast Fourier Transform", *Communications of the ACM*, Vol. 10, No. 10, October, 1967, pp. 647-654.

Sobel, Irwin [1970], "Camera Models and Machine Perception", Stanford Artificial Intelligence Project Memo No. 121.

VanLehn, Kurt [1973], "SAIL Users Manual", Stanford Artificial Intelligence Project Memo No. 204.

Yakimovsky, Yoram [1973], "Scene Analysis Using a Semantic Base for Region Growing", Stanford Artificial Intelligence Project Memo No. 209.