Stanford Artificial Intelligence Laboratory
Memo AIM-339

Department of Computer Science
Report No. STAN-CS-80-805

MODELLING THE ENVIRONMENT OF AN EXPLORING VEHICLE BY MEANS OF STEREO VISION

by

Donald B. Gennery

Research sponsored by

Advanced Research Projects Agency

COMPUTER SCIENCE DEPARTMENT
Stanford University

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) (9) Doctoral thesis

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (14) STAN-CS-80-805, AIM-339	2. GOVT ACCESSION NO. AD-A091081	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) Modelling the Environment of an Exploring Vehicle by Means of Stereo Vision.		5. TYPE OF REPORT & PERIOD COVERED technical, June 1980
		6. PERFORMING ORG. REPORT NUMBER STAN-CS-80-805 (AIM-339)
7. AUTHOR(s) (10) Donald B. Gennery		8. CONTRACT OR GRANT NUMBER(s) (15) MDA903-80-C-0102
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Computer Science Stanford University Stanford, California 94305 USA		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency Information Processing Techniques Office 1400 Wilson Avenue, Arlington, Virginia 22209		~~12. REPORT DATE~~ (11) Jun 80 / 13. NO. OF PAGES 144
		15. SECURITY CLASS. (of this report) Unclassified
14. MONITORING AGENCY NAME & ADDRESS (if diff. from Controlling Office) Mr. Philip Surra, Resident Representative Office of Naval Research, Durand 165 Stanford University (12) 154		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this report)

Approved for public release; distribution unlimited.

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

18. SUPPLEMENTARY NOTES

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)

(see other side)

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EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

094120 FW

19. KEY WORDS (Continued)

20. ABSTRACT (Continued)

This dissertation describes research involving vision techniques which would be useful in an autonomous exploring vehicle, such as a Mars rover. These techniques produce a description of the surroundings of the vehicle in terms of the position, size, and approximate shape of objects, and can match such scene descriptions with others previously produced. The information produced is thus useful both for navigation and obstacle avoidance. The techniques operate by using three-dimensional data which they can produce by means of stereo vision from stereo picture pairs or which can be obtained from a laser rangefinder. The research thus divides conveniently into two portions: stereo mapping and three-dimensional modelling and matching.

The stereo mapping techniques are designed to be suitable for the kind of pictures that a Mars rover might obtain and to produce the kind of data that the modelling techniques need. These stereo techniques are based upon area correlation and produce a depth map of the scene. Emphasis is placed upon extraction of useful data from noisy pictures and upon the estimation of the accuracy of the data produced. Included are the following: a self-calibration method for computing the stereo camera model (the relative position and orientation of the two camera positions); a high-resolution stereo correlator for producing accurate matches with accuracy and confidence estimates, which includes the ability to compensate for brightness and contrast changes between the pictures; a search technique for using the correlator to produce a dense sampling of matched points for a pair of pictures; and the computation of the distances to the matched points, including the propagation of the accuracy estimates.

The three-dimensional modelling and matching techniques are designed to be tolerant of the errors that stereo mapping techniques often produce. First, a ground surface finder tries to find a set of points that form a well-defined smooth surface that lies below most of the other points. Then, by using this knowledge of the ground surface and knowledge of the camera viewpoint that produced the points in the scene, an object finder approximates the objects that are above the ground by ellipsoids. Finally, a scene matcher can use the descriptions of scenes in terms of ellipsoidal objects. By using a search pruned by using probabilities obtained by means of Bayes' theorem, it determines the probability that two scene descriptions refer to the same scene and the linear transformation needed to bring the two scenes into alignment.

These techniques have been tried on stereo pictures of the Martian surface taken by the Viking Lander 1. The object finder was able to locate rocks fairly successfully, and the scene matcher was able to match successfully the resulting scene descriptions. Examples of these results are shown.

MODELLING THE ENVIRONMENT OF AN EXPLORING VEHICLE BY MEANS OF STEREO VISION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Donald B. Gennery
June 1980

ABSTRACT

This dissertation describes research involving vision techniques which would be useful in an autonomous exploring vehicle, such as a Mars rover. These techniques produce a description of the surroundings of the vehicle in terms of the position, size, and approximate shape of objects, and can match such scene descriptions with others previously produced. The information produced is thus useful both for navigation and obstacle avoidance. The techniques operate by using three-dimensional data which they can produce by means of stereo vision from stereo picture pairs or which can be obtained from a laser rangefinder. The research thus divides conveniently into two portions: stereo mapping and three-dimensional modelling and matching.

The stereo mapping techniques are designed to be suitable for the kind of pictures that a Mars rover might obtain and to produce the kind of data that the modelling techniques need. These stereo techniques are based upon area correlation and produce a depth map of the scene. Emphasis is placed upon extraction of useful data from noisy pictures and upon the estimation of the accuracy of the data produced. Included are the following: a self-calibration method for computing the stereo camera model (the relative position and orientation of the two camera positions); a high-resolution stereo correlator for producing accurate matches with accuracy and confidence estimates, which includes the ability to compensate for brightness and contrast changes between the pictures; a search technique for using the correlator to produce a dense sampling of matched points for a pair of pictures; and the computation of the distances to the matched points, including the propagation of the accuracy estimates.

The three-dimensional modelling and matching techniques are designed to be tolerant of the errors that stereo mapping techniques often produce. First, a ground surface finder tries to find a set of points that form a well-defined smooth surface that lies below most of the other points. Then, by using this knowledge of the ground surface and knowledge of the camera viewpoint that produced the points in the scene, an object finder approximates the objects that are above the ground by ellipsoids. Finally, a scene matcher can use the descriptions of scenes in terms of ellipsoidal objects. By using a search pruned by using probabilities obtained by means of Bayes' theorem, it determines the probability that two scene descriptions refer to the same scene and the linear transformation needed to bring the two scenes into alignment.

These techniques have been tried on stereo pictures of the Martian surface taken by the Viking Lander 1. The object finder was able to locate rocks fairly successfully, and the scene matcher was able to match successfully the resulting scene descriptions. Examples of these results are shown.

ACKNOWLEDGMENTS

There are may people who have had an influence on this thesis.

Duane Brown's famous Report 39 at RCA first introduced me to the matrix treatment of the general problem of least-squares. John Stephenson of RCA further elucidated some of the properties of least-squares adjustments. Lynn Quam and Marsha Jo Hannah first started me thinking about stereo camera models.

Elliott Levinthal and Sid Liebes provided the Mars pictures used in this research, and Mike Farmwald converted the pictures to our format.

The MACSYMA program at MIT was used in the derivation of some of the equations in this thesis, doing complicated algrebraic manipulation automatically. The thesis itself was written using the POX document compiler produced by Robert Maas. POX macros produced by Bill Gosper were used in writing the mathematical expressions. The reading committee consisting of Tom Binford, Elliott Levinthal, and John McCarthy checked the thesis.

The research described in this thesis was made possible by the fine facilities and atmosphere at the old Stanford Artificial Intelligence Lab, which, alas, no longer exists.

Most of all, I would like to thank Hans Moravec. I have used his picture processing and graphics routines used extensively. All of the figures in this thesis were produced using his graphics routines. He provided helpful advice on many matters, and by providing stimulating discussion on many topics helped to make my stay at Stanford pleasant.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Chapter 1
INTRODUCTION

This chapter describes the types of problems attacked in the current work, provides an overview of the system which was produced, and describes the notation used herein.

1.1 Motivation and Scope

An important capability possessed by all higher animals is the ability to find their way around in the world. This ability includes both obstacle detection and navigation. Although some insects rely mostly on touch and chemical sensors for this purpose, the more intelligent animals rely largely on vision, which has the advantages of long range, speed in determining shape information, and the ability to determine the reflectance of surfaces to aid in identification.

It will become increasingly important for machines to have similar capabilities. For example, a robot vehicle for planetary exploration, such as a Mars rover, should have some such ability. Because of the long radio propagation times involved (between 6 minutes and 45 minutes from Earth to Mars or between 133 minutes and 184 minutes from Earth to Titan, round trip), and because of the fact that radio transmissions may be interrupted when the vehicle is on the other side of the planet, it is highly desirable that the vehicle be largely on its own, with instructions being sent occasionally from Earth.

As with animals, so also with machines it seems desirable to rely heavily on vision. A robot vehicle may have other navigation devices which are far superior to those of any animal, such as an inertial navigator, radio navigation equipment, a wheel-revolution counter for dead reckoning, or even celestial navigation equipment, but each of these has inherent limitations (and of course is useless for obstacle detection). For example, dead reckoning can be very accurate over short distances, but as the vehicle travels errors build up without bound, and thus the position information must be corrected with occasional fixes from another source. Thus, determining the vehicle position by inspection of its surroundings when it is in a familiar area can be very important.

As a vehicle explores, it can build up a description of its environment from visual data. The vehicle position information that it needs for this purpose may come primarily from dead reckoning, perhaps supplemented by other navigation systems. However, when the vehicle enters a previously explored area, a considerable position error may have occured since its previous time there. By visual inspection of its environment it should be able to recognize the area and correct its position data.

Several types of vision systems are possible. In the most common type, one or more cameras simply passively view a scene. Although three-dimensional information can be deduced from such pictures in many cases, more direct, active means of measuring range to the points in a scene are sometimes used. One possibility is to illuminate the scene with a light source at one position and view the result with a camera at a different position. By triangulation, distance can be unambiguously determined. By scanning, the entire scene can be measured in this manner. Such a system is described by Agin and Binford [1973]. Another possibility is to measure range by the round trip time of flight of a beam of light which illuminates the scene. (The term "lidar" is sometimes used for such a device.) A raster scan can cover the entire scene. Such a system is described by Lewis and Johnston [1977]. Although not absolutely necessary, lasers are very convenient for the light source for both types of rangefinder systems and are used in the existing systems. Thus the term "laser rangefinder" is used to describe both types of system. It is a moot point whether such systems should even be called vision. It might be argued that such a laser rangefinder system is really using touch; its feelers are merely made of photons instead of solid matter. Be that as it may, the term "vision" will be used here to include such a system, not only because it uses light, but also because of the similarity of the data that it produces to the data which can be extracted from other types of vision systems, as described below. (No animal has a system exactly like this, although the sonar of porpoises and bats may come close.)

A vision system used for navigation can measure various properties of the scenes being viewed. Reflectivity patterns on objects can be used to identify them as particular landmarks, or the scene can be identified by the three-dimensional shape of its contents, regardless of their coloring. Animals use both of these methods, and so should a good robot vision system. However, the research in this thesis is restricted to the latter method.

There are basically three ways in which the desired three-dimensional information can be obtained by vision. First, a passive monocular view can be used, from which depth information can be deduced by various clues, such as perspective, shading, shadows, texture variation, and knowledge about the objects in the scene. Some of the recent work on vision systems of this type is described in Hanson and Riseman [1978] and Shirai [1978]. Second, stereo vision can be used, in which two or more views from different locations can be compared to deduce the distances to points in the scene by triangulation. This can be accomplished either by binocular vision, in which two eyes or cameras mounted in fixed positions obtain simultaneous views, or by motion parallax, in which similar information can be derived from a single moving sensor. These are equivalent if nothing moves in the scene, so they are both refered to here as "stereo." Third, a scanning laser rangefinder can be used. Animals use both monocular clues and motion parallax, and some use binocular vision also. The easiest method to use in a robot from the viewpoint of computational difficulty is the laser rangefinder. The most difficult is monocular vision, because of the inherent ambiguities in a monocular view, which must be resolved by various heuristics. Therefore, although an intelligent robot should use a combination of these methods, the use of monocular cues will not be discussed in this thesis.

If stereo vision is used, each stereo pair of pictures can be processed to produce a depth map consisting of the distances to points densely spaced over the scene. This is similar to the data that a scanning laser rangefinder produces. (The stereo method usually produces less reliable ranges and more blank areas than would the laser rangefinder method, but the latter may be limited in range and in some cases is not available.) Therefore, the further processing needed is fairly independent of which method was used. This further processing, which can be called "modelling", consists of the reduction of the information from one or more views to an abstract or symbolic form in order to save storage, to facilitate recognition upon encountering the same area in the future, and to detect obstacles.

Vision tasks may be discussed in terms of three types: description, recognition, and verification, as described by Baumgart [1974] and Bolles [1976]. The type considered in this thesis is mainly description. An unknown scene is viewed and the task is to describe it in suitable form for the data base. There will be no a priori knowledge of the precise shapes of objects in the scene, which may consist of rocks scattered around on the surface of Mars, for example. Thus the system works primarily in a bottom-up fashion, extracting from the raw data the information needed to describe the scene in terms of the approximate size, shape, and position of objects. However, some similarity to recognition and verification vision occurs when a current scene is compared to the data base.

An important issue in navigating by visual means is how, in the recognition phase, the information obtained from a view or views of the current scene is to be matched with the information previously accumulated in the data base. Should the current data be transformed into the same kind of data that is in the data base and be compared in that form, or should a portion of the data base be transformed back into a more primitive form and compared to the current scene in this low-level form? For example, if monocular vision is used, an extreme form of the second possibility would be to assume a viewing position and to project the scene into a two-dimensional picture to be compared to the actual current picture. With stereo vision or a scanning laser rangefinder, the same kind of transformation could be made, except that range pictures would be compared instead of brightness pictures.

It is an important assumption of this thesis that the best choice for the kind of scenes described above is the first of the above two possibilities. That is, the same kind of description process should be used on the current scene as was used in generating the data base. There are two principal reasons for this. First, the search needed in order to deduce position is less when the matching process operates on symbolic data. If the data were to be matched in the form of numerical values of distance or height at some dense sampling, values of vehicle position spaced at some fine increment would have to be tried in order to determine the best matching position. Second, in order to save storage and to make the stored data as independent as possible of viewing conditions, the representation used must discard a good deal of information about the scene and abstract only the significant features. It then becomes impossible to reconstruct exactly what would be seen from any particular point. Note that this conclusion might be quite different if the data base consisted of accurate models of objects, such as known manmade objects, in which case a top-down recognition process might be more appropriate.

One purpose of the research described herein is to develop a way of representing scenes so that they can be matched in this manner and a technique for matching them, and to see how well these methods work on typical outdoor scenes such as might be seen by a Mars rover. This problem will be discussed further in Section 1.3. Another purpose is to develop stereo vision techniques which produce the kind of data that these methods need. The rationale behind these techniques and their general outline will be discussed in Section 1.2.

1.2 Stereo Processing

Several different types of stereo vision systems are possible, depending on the level at which the matching of the two (or more) pictures occurs. Area correlation operates on the brightness levels in the actual pictures, attempting to match a small window in one picture to some area of the same size in the other picture. (Hannah [1974] and many others have used this method.) Edge correlation first applies an edge operator to the pictures to detect brightness edge elements and then attempts to match the edges in one picture to those in the other picture, as in Arnold [1978]. Other techniques match even higher features extracted from the pictures (for example Ganapathy [1975]). Each of these methods has advantages and disadvantages, and for different types of pictures different methods may be most suitable. (Stereo vision systems also can differ in the number of views used to extract the depth information. For example, Moravec [1979 and 1980] uses nine-eyed stereo to aid in resolving ambiguities. However, only two-eyed stereo is used in the system described in this thesis.)

Scenes of manmade objects often contain smooth lines of high contrast separating regions that may be fairly uniform in brightness. In this case a system based on edges or lines would be appropriate. The edge detector would be able to locate the boundaries in the scene accurately. An area correlator might not be able to match at an edge produced by a depth discontinuity if the background side of the edge is textured, and if the foreground object is untextured the edges might be the only information about it that can be seen.

On the other hand, natural outdoor scenes often are highly textured. An edge detector might produce an enormous number of edges to deal with, and the boundaries of objects may be very rough and thus not produce edges that can be easily matched. However, area correlation should work well in this case. The presence of texture within the windows being matched should produce good matches over most of the scene. Trouble may still exists at edges of objects because of depth discontinuities, but if matches are found over the rest of each object, these may suffice.

The use of area correlation produces a loss in resolution, because the match represents an average over the match window. Typically the match window may be eight pixels by eight pixels. in which case there would by a loss of resolution of a factor of eight. If highly detailed information about the shape of objects is desired, this would be a drawback. However, the application in mind here is to produce data for the modelling process, which will discard fine details anyway in order to represent the scene economically in a partially symbolic form. Therefore, the loss of resolution is not very harmful here.

For these reasons area correlation will be used in this thesis. It should produce points for which the distance can be calculated spread over the ground and over the surface of each large object in the scene. From this information the objects can be detected and their approximate size and shape can be measured in the modelling process.

The particular type of correlator used in this research, which includes some improvements over usual cross correlation, is described in Chapter 2. It produces a match when applied locally to a small area by some higher-level procedure, which must perform the more global search. In addition to the computed most probable position of match in the image plane, it computes a two-by-two covariance matrix which represents the estimated accuracy of the match, and a probability estimate which indicates the goodness of the match. The correlator includes the ability to allow for various amounts of brightness and contrast change between the pictures, depending on the available knowledge about the pictures.

In some cases the stereo camera model (the relative position and orientation of the cameras which produced the stereo views) is accurately known before the pictures are obtained. For example, two cameras may be rigidly mounted on a vehicle and their positions and orientations may have been accurately measured in the laboratory. In other cases, the stereo camera model may be unknown or inaccurately known. For example, flexure in a vehicle may cause slight variations from the previously measured values, or a one-camera vehicle may move to separate, poorly known, locations for the individual pictures. In such cases the information to calibrate the stereo camera model can come from the pictures themselves, although the distance between the cameras cannot be so determined. Such a self-calibration method is described in Chapter 3. This involves finding a set of matching points sparsely scattered over the picture, applying the correlator to these points, and using the resulting information to solve for the parameters that define the position and orientation of one camera relative to the other.

In any case, once the stereo camera model is known it is easier to produce a dense sampling of matched points over the pictures, because the necessary search is one-dimensional instead of two-dimensional. A search procedure for doing this, which applies the correlator locally as needed, is described in Chapter 4. It decides whether or not to accept matches by using the probability values produced by the correlator and the agreement of matches with neighboring matches.

From each matched pair of points found by this search procedure, the distance to the corresponding point in the scene (and thus the coordinates of a point in three-dimensional space) can be computed by using the stereo camera model. The computations for doing this, including the use of the accuracy estimates, are described in Chapter 5.

The mathematics derived in this thesis for the above computations assume that the relationship between points in three-dimensional space and points in the image plane is the central projection. (See, for example, Duda and Hart [1973].) Where this is not the case, distortion corrections can be included in the computations to convert between the central projection and the actual projection used, insofar as it is known. The places where this occur in the processing are pointed out in the subsequent chapters. Two types of distortion correction are available in the implemented programs. One of these assumes that the image pixel coordinates, together with range, form a spherical coordinate system, as is the case with the Viking Lander pictures. The other uses a two-dimensional polynomial to describe the distortion. It is used with pictures taken with the Stanford AI Lab Cart, and a way of calibrating this distortion is described by Moravec [1979 and 1980].

1.3 Three-Dimensional Modelling and Matching

Regardless of whether a scanning laser rangefinder or stereo vision has been used, the result is the three-dimensional coordinates of a set of points in the scene. This must be converted into a suitable form for storing in the data base of an exploring vehicle and for comparing to previously accumulated information in the data base.

One possibility would be simply to express each point in terms of height as a function of horizontal position, relative to a nominally horizontal reference plane. Points gathered from several observations would simply be combined. When a scene described in this fashion is compared to similar data in the data base, the heights would be correlated using something similar to the ordinary two-dimensional correlation coefficient, except that it would have to take into account the fact that the points are not equally spaced (there even may be large blank areas). There are several disadvantages to this method. The data base would require a large amount of storage, and a search to find the correct match would require a large amount of computing. Therefore, the scene should be represented in a more compact, abstract way.

The way that has been chosen here represents a scene in terms of ellipsoidal objects and a ground surface. A scene represented in this way is in the form needed for obstacle avoidance, in addition to being suitable for storing and comparison with the data base. It is not necessary that the actual objects in the scene (for example, rocks on the surface of Mars) be ellipsoids. Approximating them by ellipsoids, while throwing away a good deal of information about their shape, retains the important information needed for obstacle avoidance (unless it is desired to pass very close to obstacles), and, if there are many objects in a scene, sufficient information is retained for recognition. (Of course, some types of obstacles, such as cliffs, could not very well be represented as objects in this way. However, they could be included as part of the ground surface.)

Thus, information from one or more stereo pairs or scanning laser rangefinder views is transformed into coordinate system aligned with a nominally horizontal plane. A ground surface finder is then used to find the ground for portions of the scene, which may be tilted slightly relative to the assumed horizontal coordinate system. In addition, the computed ground surface may be curved and may have other complications, depending on the exact method used, as described in Chapter 6. The ground surface finder operates by trying to find a set of points that form a well-defined smooth surface that lies below most of the other points. It allows a few points below this surface, such as might be caused by errors in the stereo processing, and it allows a fairly large number of points above the surface, such as might occur on objects.

The next step in the three-dimensional modelling consists of the application of an object finder, described in Chapter 7. Points that are sufficiently far above the ground surface are clustered into objects approximated by ellipsoids. Each ellipsoid is adjusted in such a way so as not only to fit the points which seem to lie on this object but also to avoid hiding other points as seen from the camera position.

The representation of the scenes used in the data base could use both the ellipsoid information (their positions, sizes, and shapes) and some characteristics of the ground computed by the ground surface finder (perhaps slope, curvature, and discontinuities). However, only the use of the ellipsoid information has been implemented in a scene matcher. For fairly flat ground covered with many objects such as rocks, this is the most important information.

The scene matcher compares the descriptions of two scenes in terms of ellipsoidal objects, as described in Chapter 8. By using Bayes' theorem it determines the probability that two scene descriptions refer to the same scene and the translation needed to bring the two scenes into alignment. It also can adjust for small rotations and scale factor changes.

The techniques described in this thesis have been tested on pictures primarily from two sources. One of these is the old version of the Stanford AI Lab Cart described by Moravec [1977]. The object finder was able to locate cars in a parking lot using pictures digitized from the television camera on the Cart. The other source is the Viking Lander 1 on the surface of Mars. Sample results using these Mars pictures are given in appropriate places in this thesis. The Mars pictures used for this purpose are described in Appendix C.

1.4 Notation

Matrix notation is used heavily in this thesis. A reader who is unfamiliar with matrix algebra can find the necessary background information in Hohn [1973].

Matrices are denoted here by capital letters and scalars by lower-case letters, with the exceptions mentioned below. The transpose of a matrix A is denoted by A^{T}, and the inverse of A is denoted by A^{-1}. The trace of a square matrix A (sum of the diagonal elements, which is equal to the sum of the eigenvalues) is denoted by tr(A), and the determinant of A is denoted by det(A). The identity matrix of any size is denoted by I. The term "vector" is used here in general to denote any column matrix.

In the special case of a physical vector in three-dimensional space, the vector is represented by a 3-by-1 matrix containing the components in a particular rectangular coordinate system. However, in this case the symbol for the vector will be a boldface lower-case letter instead of a capital letter, to emphasize its nature. The cross product of two physical vectors **a** and **b** is denoted by $\mathbf{a} \times \mathbf{b}$. The magnitude ($=\sqrt{\mathbf{a}^{\mathrm{T}}\mathbf{a}}$) of a vector **a** is denoted by a or $|\mathbf{a}|$. A unit vector is denoted by the symbol $\mathbf{1}$ with an appropriate subscript. (Thus, $\mathbf{a} = a\mathbf{1}_a$, where **a** represents any vector, provided that **a** and $\mathbf{1}_a$ are expressed in the same coordinate system.)

It sometimes will be needed to deal with derivatives involving matrices. The derivative of a matrix (including the case of a vector) with respect to a scalar is defined to mean the matrix whose elements are the derivatives of the elements of the original matrix with respect to the scalar. The derivative of a scalar with respect to a vector (column matrix) is defined to mean the row matrix whose elements are the partial derivatives of the scalar with respect to the elements of the vector. The derivative of a vector with respect to another vector is defined to mean the matrix composed of the partial derivatives of the individual elements, such that the rows correspond to the elements of the first vector and the columns correspond to the elements of the second vector. (That is, the i,j element of $\frac{dA}{dB}$ is $\frac{\partial a_i}{\partial b_j}$.) Other combinations are not defined.

Standard mathematical symbols are used. Thus the use of the symbol Γ to denote the gamma function and the references to the F test in statistics represent exceptions to the above rule about capital letters being used for matrices.

Symbols for individual quantities differ from chapter to chapter and will be defined as needed.

Chapter 2

STEREO CORRELATOR

This chapter describes the correlator (called the high-resolution correlator in Gennery [1977]) which refines local matches between a pair of pictures. It is the basic low-level component which operates on raw picture data and produces the information used by all the higher-level components of the system described in this thesis (except in the camera model solution, if done, where Moravec's interest operator and binary-search correlator also operate on raw data, as described in Chapter 3).

2.1 Statement of Problem

Consider the following problem. A pair of stereo pictures is available. For a given point in Picture 1, it is desired to find the corresponding point in Picture 2. It will be assumed here that a higher-level process has found a tentative approximate matching point in Picture 2, and that there is an area surrounding this point, called the search window, in which the correct matching point can be assumed to lie. A certain area surrounding the given point in Picture 1, called the match window, will be used to match against corresponding areas in Picture 2, with their centers displaced by various amounts within the search window in order to obtain the best match.

Let $a_1(x,y)$ represent the measured brightness values in Picture 1, $a_2(x,y)$ represent the measured brightness values in Picture 2, x_1, y_1 represent the point in Picture 1 that we desire to match, x_2', y_2' represent the center of the search window in Picture 2, w_m represent the width of the match window (assumed to be square), and w_s represent the width of the search window (assumed to be square), where x and y take on only integer values representing individual pixels.

The following assumptions are made. The pixel values a_1 and a_2 consist of true brightness values linearly related to each other, translated by an unknown amount in x and y, and having normally distributed random errors added. The errors are uncorrelated with each other, both within a picture (autocorrelation) and between pictures (cross correlation), and the errors are uncorrelated with the true brightness values. (The assumptions concerning errors hold fairly accurately for the usual noise content of pictures. However, another type of change is perspective distortion, which can be important with large match windows, but it will not be discussed here.) No assumptions about the nature of the true picture content are made, except briefly when discussing interpolation in Section 2.5.

Thus the assumed relationship between the measured brightness values in the two pictures is

$$a_2(x+x_c-x_1, y+y_c-y_1) + r_2(x,y) = b\sqrt{1+c^2} + ca_1(x,y) + r_1(x,y) \qquad (2.1\text{-}1)$$

where x_c, y_c is the true matching point in Picture 2 corresponding to x_1, y_1 in Picture 1, $r_1(x,y)$ and $r_2(x,y)$ are independent normally distributed random variables (noise), and b and c are the brightness and contrast changes (bias and scale factor) between the pictures. (The factor $\sqrt{1+c^2}$ is included in the bias term so that the bias represents the perpendicular distance from the origin in a_1, a_2 space to the straight line with slope of c which represents the relationship between a_1 and a_2. This makes the relationship symmetrical with respect to interchanging a_1 and a_2.)

It is further assumed that a priori values of bias and scale factor b_0 and c_0 and their standard deviations σ_{b_0} and σ_{b_0} are available. There may also be some information available about the standard deviation of the noise, as described in Section 2.3.

The correlator should use the information in the pictures (the portions specified by the windows) and whatever information is available about bias, scale factor, and noise in order to arrive at an estimate of the matching point x_c, y_c, suppressing the noise as much as possible based on the statistics of the noise. It also should produce an estimate of the accuracy of the match in the form of the variances and covariance of the x and y coordinates of the matching point in the second picture, and an estimate of the probability that the match is consistent with the statistics of the noise in the pictures, rather than being an erroneous match. The subsequent sections explain how these goals are achieved.

2.2 Basic Correlator

It is assumed in this section that the standard deviation of the noise is known for every point in each picture, that the bias and scale factor are known to be zero and unity, respectively, and that x_c and y_c are integers (that is, no fractional-pixel shifts have occured).

Thus we now wish to find the matching point x_m, y_m which will produce the best match of $a_2(x+x_m-x_1, y+y_m-y_1)$ to $a_1(x,y)$ in some sense. Traditionally the match which maximized the correlation coefficient between a_1 and a_2 has been used (as in Hannah [1974]). Indeed, this is optimum when the bias and scale factor are completely unknown, if one of the two functions has no noise. However, here both functions have noise. This fact introduces fluctuations in the cross-correlation function which may cause its peak to differ from the expected value. Ad hoc smoothing techniques could be used to reduce this effect, but an optimum solution can be derived from the assumed statistics of the noise.

Let E represent the w_m^2-vector of the differences $a_2(x+x_m-x_1, y+y_m-y_1) - a_1(x,y)$ over the w_m-by-w_m match window, for a given trial value of x_m, y_m, and let x_c, y_c represent the true (unknown) value of x_m, y_m. Let p represent a probability and ρ represent a probability density with respect to the vector E. Then by Bayes' theorem

$$p(x_m, y_m = x_c, y_c \mid E) = \frac{p(x_m, y_m = x_c, y_c)\, \rho(E \mid x_m, y_m = x_c, y_c)}{\Sigma\, p(x_m, y_m = x_c, y_c)\, \rho(E \mid x_m, y_m = x_c, y_c)} \tag{2.2-1}$$

If we assume that the a priori probability $p(x_m, y_m = x_c, y_c)$ is constant over the search window and is zero elsewhere, this reduces to

$$p(x_m, y_m = x_c, y_c \mid E) \propto \rho(E \mid x_m, y_m = x_c, y_c) \tag{2.2-2}$$

Since E consists of uncorrelated normally distributed random variables,

$$\begin{aligned} \rho(E \mid x_m, y_m = x_c, y_c) &\propto \prod \exp\left(-\frac{e_i^2}{2(\sigma_1^2 + \sigma_2^2)}\right) \\ &= \exp\left(-\frac{1}{2} \sum \frac{e_i^2}{\sigma_1^2 + \sigma_2^2}\right) \end{aligned} \tag{2.2-3}$$

where e_i denotes the components of E, σ_1 and σ_2 are the standard deviations of a_1 and a_2, and the product and sum are taken over the match window. (Very often, the variances σ_1^2 and σ_2^2 can be considered to be constant. In this case, the summation can be reduced to the sum of the squares of the differences over the match window, with the sum of the two variances factored out.) Thus, defining w to be

$$w = \exp\left(-\frac{1}{2} \sum \frac{e_i^2}{\sigma_1^2 + \sigma_2^2}\right) \tag{2.2-4}$$

produces

$$p(x_m, y_m = x_c, y_c \mid E) = kw \tag{2.2-5}$$

where k is a constant of proportionality.

So far, the derivation is quite usual. If we simply wanted to maximize p (for the maximum likelihood solution), we would minimize the above sum (that is, use a weighted least-squares solution). However, because of the fluctuations in w caused by the presence of noise in both images, the peak of p in general differs from the center of the distribution of p in a random way due to the random nature of the errors.

Therefore, we define the optimum estimate of the matching position to be the mathematical expectation of x_m, y_m according to the above probability distribution. Thus, letting (x_2, y_2) represent this optimum estimate, we have

$$x_2 = \frac{\Sigma\, w x_m}{\Sigma\, w}$$
$$y_2 = \frac{\Sigma\, w y_m}{\Sigma\, w} \qquad (2.2\text{-}6)$$

where the sums are taken over the search window. The variances and covariance of x_2 and y_2 are given by the second moments of the distribution around the expected values:

$$\sigma^2_{x_2} = \frac{\Sigma\, w x_m^2}{\Sigma\, w} - x_2^2$$
$$\sigma^2_{y_2} = \frac{\Sigma\, w y_m^2}{\Sigma\, w} - y_2^2 \qquad (2.2\text{-}7)$$
$$\sigma_{x_2 y_2} = \frac{\Sigma\, w x_m y_m}{\Sigma\, w} - x_2 y_2$$

The covariance matrix of x_2 and y_2 consists of $\sigma^2_{x_2}$ and $\sigma^2_{y_2}$ on the main diagonal and $\sigma_{x_2 y_2}$ on both sides off the diagonal.

Because of the finite search window size, the covariance matrix computed by (2.2-7) may be an under-estimate. It is possible to apply an approximate correction for this effect (and the implemented correlator does so), but as long as the width of the correlation peak represented by (2.2-5) is considerably less than the width of the search window, the effect is negligible.

It might appear that the above analysis is not correct because of the fact that certain combinations of errors at each point of each picture are possible for more than one match position, and the probability of these combinations is split up among these match positions. However, this fact does not influence the results, as can be seen from the following reasoning. The possible errors at each point of each picture form a multidimensional space. When a particular match position is chosen, a lower-dimensioned subspace of this space is selected, in order to be consistent with the measured brightness values. When another match is chosen, a different subspace is selected. These two subspaces in general intersect, if at all, in a subspace of an even lower number of dimensions. Thus the hypervolume (in the higher subspace) of this lower subspace is zero. Therefore, the fact that the two subspaces intersect does not change the computed probabilities.

The computation of the output probability estimate depends on some quantities computed in the variance estimation portion of the computations, so it will be discussed in the next section.

2.3 Variance Estimation

If the amount of noise in the pictures is not known, it is possible to estimate it from the pictures themselves, provided that some assumptions are made about the way in which the variance (square of the standard deviation) of the noise varies over the pictures. In some cases, the variance may be a function of brightness. For example, shot noise variance, which is the primary source of noise with some photodiodes, is proportional to the amplitude of the signal (and thus the standard deviation is proportional to the square root of the signal). For some solid-state cameras the noise might need to be calibrated for each pixel in the picture. (If it can be completely calibrated, then it can be used in the equations in Section 2.2, and no variance adjustment is needed.) It is assumed in this section that the noise variance is constant over each picture. (If it actually varies with brightness in a known way, the data can be transformed by a nonlinear function to make the variance constant. For example, taking the square root of each pixel brightness value will cause shot noise to become constant.)

Let v represent the total variance in both pictures. Thus

$$v = \sigma_1^2 + \sigma_2^2 \qquad (2.3\text{-}1)$$

The task at hand is to estimate v, since this is the quantity that is needed in (2.2-4).

Often some knowledge is available about the noise variance, even if it is not known exactly. It is assumed here that this knowledge can be represented by a chi-square distribution. Let the a priori value of v be denoted by v_0. Then it is assumed that v_0 has the chi-square distribution with n_0 degrees of freedom. This assumption is made both for its convenience in the subsequent calculations and because of the fact that, if the variance has been estimated by squaring n_0 samples from a normal distribution and averaging them, it will have this distribution. Thus n_0 can be considered to be the weight of the observation v_0. (See Hogg and Craig [1965] for this and other information about the chi-square distribution.) If the variance is completely unknown, $n_0 = 0$. If it is known exactly, $n_0 = \infty$.

An estimate of the variance can be obtained from the goodness of fit between the two pictures when matched. This computed estimate is denoted v_c. If the correct matching point x_c, y_c were known, it could be used for x_m, y_m to compute the vector of differences between the two pictures E, and then a good value for v_c would be the mean square value of these differences, $\Sigma e_i^2/w_m^2$. However, the correct match is not known. However, w from (2.2-4) is proportional to the probability that each x_m, y_m match is correct, according to (2.2-5). Therefore, a weighted average over the search window, with w as the weight, of the mean (over the match window) squared value of the differences is used as a preliminary value for v_c. That is,

$$v_c' = \frac{\Sigma\,(w\,\Sigma\,e_i^2)}{w_m^2\,\Sigma\,w} \tag{2.3-2}$$

where the outer sums are taken over the search window and the inner sum in the numerator is taken over the match window. (Remember that w and the differences e_i are implicit functions of the position of x_m, y_m within the search window, in addition to e_i being a function of the position within the match window as indicated explicitly by the subscript i.) Since the computation of w requires the value of v in (2.2-4), the process is iterative, and v from the previous iteration is used to compute w in this iteration.

The estimate given by (2.3-2) is called "preliminary" because the process of averaging over the search window, weighted by w, introduces a bias. The mean squared residuals fluctuate over the search window because of the random nature of the noise in the pictures. The weights are computed from this value according to (2.2-4), and thus there is a statistical tendency for the smaller values to have the greater weights. This causes v_c' to be an underestimate of the variance by varying amounts depending on the sharpness of the correlation peak. At one extreme, the correlation peak is very sharp, one value of w is much larger than all of the others, and thus only one term has any appreciable effect in the summation over the search window, and since this is almost certainly the correct matching point there is no bias in v_c'. At the other extreme, when the correlation peak is very broad, there are many roughly equal terms in the summation, with fluctuations from noise greater than their difference from the peak caused by true brightness differences across the pictures. In this case v_c' approaches being an underestimate by some constant factor.

To see how much of an underestimate is produced in the limiting case of a very broad correlation peak, first note that the sum of the squares of the differences has the chi-square distribution with w_m^2 degrees of freedom (because the noise has the normal distribution). Specifically, if we define

$$u = \frac{\Sigma\,e_i^2}{v} \qquad n = w_m^2 \tag{2.3-3}$$

where v is the true total variance, then the probability density function of u is

$$\rho = \frac{1}{\Gamma(\frac{n}{2})2^{n/2}}\,u^{n/2-1}\exp(-\frac{u}{2}) \tag{2.3-4}$$

Now assume that the variance is known exactly, so that the correct variance is used in (2.2-4), which becomes

$$w = \exp(-\frac{u}{2}) \tag{2.3-5}$$

(The variance may not be known at the start, of course, but as the method iterates to a final solution for variance, a good estimate of the variance will become available to compute w.) Then (2.3-2) can be rewritten as

$$v_c' = \frac{v \sum wu}{n \sum w} \tag{2.3-6}$$

In the limit as the search window over which the summations are taken increases without limit, the ratio in (2.3-6) approaches a constant that is the ratio of the expected values of the quantities wu and u. Thus,

$$v_c' = \frac{v \int_0^\infty \rho w u du}{n \int_0^\infty \rho w du} \tag{2.3-7}$$

Substituting (2.3-4) and (2.3-5) into (2.3-6) produces

$$v_c' = \frac{v \int_0^\infty \frac{1}{\Gamma(\frac{n}{2})2^{n/2}} u^{n/2-1} \exp(-u) u du}{n \int_0^\infty \frac{1}{\Gamma(\frac{n}{2})2^{n/2}} u^{n/2-1} \exp(-u) du}$$

$$= \frac{v \int_0^\infty \frac{1}{\Gamma(\frac{n}{2})2^{n/2}} (2u)^{n/2-1} \exp(-\frac{2u}{2})(2u) d(2u)}{2n \int_0^\infty \frac{1}{\Gamma(\frac{n}{2})2^{n/2}} (2u)^{n/2-1} \exp(-\frac{2u}{2}) d(2u)} \tag{2.3-8}$$

But the integrand in the last denominator is the probability density function of the chi-square distribution with n degrees of freedom and $2u$ as the variable, which integrates to unity, and the integral in the numerator similarly is the expected value of the chi-square distribution with n degrees of freedom, which is n. Thus (2.3-8) simplifies to

$$v_c' = \frac{v}{2} \tag{2.3-9}$$

This means that in this extreme case the variance computed by (2.3-2) is only half as large as it should be.

Thus v_c' is too small by a factor that can be anywhere from 0.5 to 1. Since this range is so small and since variances are seldom known very accurately anyway, it is possible to adequately correct for this bias by an empirical formula. A way of determining approximately where we are between these extremes is to examine the minimum value of $\Sigma e_i^2/w_m^2$ over the search window, denoted by v_m. Because this is the quantity which is averaged to produce v_c', if the correlation peak is sharp v_m is approximately equal to v_c'. But v_m tends to become smaller and smaller relative to v_c' as the correlation peak becomes broader, because of the statistical fluctuations. By simulating a variety of cases with pseudo-random numbers, an approximate correction factor based on the ratio v_m/v_c' was determined. It is applied to the variance estimate as follows:

$$v_c = \frac{v_c'}{1 - 0.5\,(1 - \frac{v_m}{v_c'})^{0.5}} \qquad (2.3\text{-}10)$$

where v_m is the minimum value of $\Sigma e_i^2/w_m^2$ over the search window, as defined above.

A weight, or accuracy estimate, must be assigned to the above variance estimate so that it can be combined with the other sources of information. The exactly correct weight to use would be difficult to determine. A conservative approximation which is adopted is to consider v_c to have the chi-square distribution with n_c degrees of freedom, where n_c is w_m^2-2 or 200, whichever is less. (Thus n_c is the weight.) The reasons for this choice are that the mean squared differences would have the chi-square distribution with w_m^2 degrees of freedom at the correct matching point, two degrees of freedom are subtracted to allow for the two degrees of freedom that are involved in adjusting the position in the image plane, the averaging over the search window increases the weight somewhat but in a way that is hard to estimate (because of the duplication of data influencing the average), and the approximate nature of the correction factor introduces some additional uncertainty, for which the limit of 200 is included.

A third source of information about the noise in the pictures can be obtained from their high-frequency content. This produces only an estimate of an upper limit to the variance, because the high-frequencies may contain true picture information in addition to noise. However, the high spatial frequencies are better to use for this purpose than any other frequency band, because picture content usually tends to be concentrated at the lower frequencies. First, the square of the output of a simple two-dimensional high-pass filter is computed as follows for each picture:

$$U = \frac{[a(x-1,y) + a(x+1,y) + a(x,y-1) + a(x,y+1) - 4\,a(x,y)]^2}{20} \qquad (2.3\text{-}11)$$

Then U is averaged over the match window in each picture and the results for the two pictures are added together to form the estimate of the upper limit of v, denoted v_u. The weight assigned to this estimate is $n_u = 2w_m^2$, because this is the number of observations which are averaged to produce the estimate.

Thus there are three estimates of noise variance, v_0, v_c, and v_u, with weights n_0, n_c, and n_u, which result from the a priori values, goodness of fit, and high-frequency content, respectively. These must be combined to produce an overall estimate of variance v, and must be compared to produce the probability estimate. (Some of the above formulas for these quantities will be altered slightly in subsequent sections.)

If the estimate of v on the current iteration is less than v_u, the value of v_u does not matter, since it is only an upper limit. Therefore, in this case the new estimate of v is the weighted average of v_0 and v_c, as follows:

$$v = \frac{n_0 v_0 + n_c v_c}{n_0 + n_c} \qquad (2.3\text{-}12)$$

On the other hand, if the current estimate of v is greater than v_u, all three values are averaged, as follows:

$$v = \frac{n_0 v_0 + n_u v_u + n_c v_c}{n_0 + n_u + n_c} \qquad (2.3\text{-}13)$$

The iterative process for v as described above undergoes linear convergence, and in some cases it converges rather slowly. Therefore, convergence acceleration is applied to it, using a one-dimensional special case of the acceleration method described in Appendix A, which is equivalent to Aitken's extrapolation (see Acton [1970]).

The probability estimate is derived by comparing the estimate of noise variance obtained from the goodness of fit (v_c) to the other estimates. Since it is assumed here that each of these estimates has a chi-square distribution, the Snedecor-Fisher F test is the appropriate way to do this. (See Hogg and Craig [1965]). If the value of v_c on the last iteration is less than v_u, the value of v_u does not matter, approximately. Therefore, in this case the quantity computed is the probability that the ratio of the variance of a sample with n_c degrees of freedom to the variance of a sample with n_0 degrees of freedom, both from the same distribution, will exceed v_c/v_0. On the other hand, if the final value of v_c is greater than v_u, both v_0 and and v_u must be considered. However, in this case the distribution of the combined v_0 and v_u values is not chi-square because of the fact that v_u is only an upper limit. As an approximation, the lesser of two F-test probabilities is used, one as above using v_0, and the other using v_u instead, with n_u degrees of freedom.

2.4 Brightness and Contrast Adjustment

In many cases changes in the brightness of each point in a scene may occur between the two pictures of a stereo pair. Some causes are differences in the cameras that took the two pictures, directional reflectivity of the surfaces, and changes in illumination if the pictures were taken at different times. It is assumed here that these changes can be approximated within the search window of the correlator by a linear function. Thus the changes can be represented by brightness bias and contrast change (scale factor) between the two pictures.

In order to allow for change in brightness and contrast between pictures in area correlation when using ordinary cross correlation, a common approach has been to use the correlation coefficient as the quantity to be maximized, as in Hannah [1974]. The correlation coefficient is normalized so that it is invariant under a linear transformation of the brightness values. However, using such a criterion throws away a lot of information about the pictures unless the brightness and contrast changes are completely unknown, which is seldom the case. The correlator described here has the ability to incorporate the a priori knowledge about these changes in the form of the standard deviations σ_{b_0} and σ_{c_0}. If these are infinity, the changes are completely unknown, and the correlator is free to adjust them in order to obtain a good fit, as with the ordinary correlation coefficient. At the other extreme, if these standard deviations are zero, the changes are constrained completely, and the correlator accepts only an exact match between the pictures, except for noise.

The equations derived below include a weight equal to the reciprocal of the variance for each point, so that they can be used in the general case where the noise variance is not constant over the picture. If the variance is constant, it can be factored out of the summations. If the variance adjustment described in Section 2.3 is done, the variance must be assumed to be constant, and factoring out the variance avoids having to recompute things as the variance changes during the iterations. (The implemented version makes this assumption always.) The variance σ^2 to be used here is the variance in each picture, if the variances are equal in the two pictures, which is $v/2$. If the variances are not equal, what is wanted is the the component of variance perpendicular to the line with slope c in a_1, a_2 space. Letting $\theta = \arctan c$, we have

$$\sigma^2 = \sigma_1^2 \sin^2\theta + \sigma_2^2 \cos^2\theta \qquad (2.4\text{-}1)$$

(In general, this equation would also include the term $-2\sigma_{12} \sin\theta \cos\theta$, but, since we have assumed that the noise in the two pictures is uncorrelated, σ_{12} is zero.) However, the eigenvector method described below assumes that $\sigma_1 = \sigma_2$ (that is, the amount of noise in the two pictures is equal). If they are widely unequal, large departures from optimality may occur. If this is the case, one of the pictures can be rescaled to make the variances at least approximately equal. Note that equation (2.4-1) requires the use of c (to obtain θ), which has not been computed yet. In general, this could be solved by iteration, but if the variances are approximately equal, using c_0 for c here should suffice. (If $\sigma_1 = \sigma_2$ exactly, θ drops out of (2.4-1).)

First consider σ_{b_0} and σ_{c_0} to be infinity. Then at any tentative matching position x_m, y_m what is desired instead of the sum of the squares of the differences in (2.2-4) and (2.3-2) is a measure of dispersion about a linear fit between the values in a_1 and a_2, taken over the match windows. This is equivalent to fitting a straight line to points in two dimensions, where the errors occur in both coordinates of each point. As discussed by Duda and Hart [1973], the appropriate method to use in such a case is the eigenvector method. In the unweighted case, this method minimizes the sum of the squares of the perpendicular distances from the points to the fitted line, the minimum value achieved is the smaller eigenvalue of the distribution about its mean, and the direction of the line is the eigenvector corresponding to the larger eigenvalue. The measure of dispersion that is desired here is this minimized sum. (Actually, twice this is equivalent to the sum of squares of differences, because the difference between an a_1 value and an a_2 value is $\sqrt{2}$ times the distance from the corresponding point to a 45° line through the origin in a_1, a_2 space.)

In order to compute the eigenvalue desired above, the weighted moments of the distribution about the mean are first computed, as follows:

$$s_{11} = \sum \frac{1}{\sigma^2}\left(a_1 - \frac{1}{n}\sum \frac{a_1}{\sigma^2}\right)^2$$

$$s_{12} = \sum \frac{1}{\sigma^2}\left(a_1 - \frac{1}{n}\sum \frac{a_1}{\sigma^2}\right)\left(a_2 - \frac{1}{n}\sum \frac{a_2}{\sigma^2}\right) \tag{2.4-2}$$

$$s_{22} = \sum \frac{1}{\sigma^2}\left(a_2 - \frac{1}{n}\sum \frac{a_2}{\sigma^2}\right)^2$$

where

$$n = \sum \frac{1}{\sigma^2}$$

and where the summations are taken over all points in the match window, with a_1 and a_2 aligned according to the current value of x_m and y_m (that is, $a_2(x+x_m-x_1, y+y_m-y_1)$ is paired with $a_1(x,y)$). Then the smaller eigenvalue is

$$s = \frac{1}{2}\left(s_{22} + s_{11} - \sqrt{(s_{22}-s_{11})^2 + 4s_{12}^2}\right) \tag{2.4-3}$$

This eigenvalue replaces $\Sigma e_i^2/(\sigma_1^2+\sigma_2^2)$ in (2.2-4), so that instead of (2.2-4) the following is used to compute w:

$$w = \exp(-\frac{s}{2}) \tag{2.4-4}$$

If variance is being adjusted, s is multiplied by the current value of $2\sigma^2$ (assumed to be constant over the picture), which cancels out the effect of σ in these equations, to obtain the value which replaces Σe_i^2 in (2.3-2). Thus, instead of (2.3-2),

$$v_c' = \frac{2\sigma^2 \Sigma ws}{w_m^2 \Sigma w} \qquad (2.4\text{-}5)$$

Now consider the effect of the a priori knowledge about brightness bias. This knowledge can be incorporated into the solution by introducing a fictitious point into the summations in (2.4-2), which represents a point through which the fitted line would have to pass if $b = b_0$ exactly. The coordinates of this fictitious point are $a_1 = -b_0c/\sqrt{1+c^2}$ and $a_2 = b_0/\sqrt{1+c^2}$. (The value of c to be used here can be c_0 as a reasonable approximation, or the solution can be iterated using improved computed values of c on each iteration. The implemented version assumes that $b_0 = 0$, in which case a_1 and a_2 are both zero for this fictitious point.) In order for this fictitious point to have the proper amount of effect according to the assumed accuracy of b_0, it must be weighted by the reciprocal of its variance. Since the variance of this fictitious point in each dimension is $\sigma_{b_0}^2$,

$$n_{b_0} = \frac{1}{\sigma_{b_0}^2} \qquad (2.4\text{-}6)$$

The extra values of a_1 and a_2 are multiplied by n_{b_0} and added into the summations in (2.4-2), and n is increased by n_{b_0} in (2.4-2).

The a priori knowledge about contrast change can be incorporated into the solution in the following way. Consider the effect of adding two fictitious points to the solution, each at a distance r from the origin in opposite directions from the origin on a line with slope c_0 in a_1, a_2 space. Let the angle between this line and the a_1 axis be θ_0, so that $c_0 = \tan\theta_0$. (In the implemented version c_0 is assumed to be unity, and thus $\theta_0 = 45°$.) Then $a_1 = \pm r\cos\theta_0$ and $a_2 = \pm r\sin\theta_0$ for these points. The a priori accuracy can be expressed in terms of the standard deviation of the angle,

$$\sigma_{\theta_0} = \sigma_{c_0}\cos^2\theta_0 \qquad (2.4\text{-}7)$$

Let σ_0 be the assumed standard deviation of the fictitious points in each dimension. If these two fictitious points were the only points in the solution, σ_{θ_0} would represent the accuracy of the direction of the line connecting them. Since the distance between the points is $2r$, each point contributes $(\sigma_0/2r)^2$ to the variance of the angle, so that the effect of both points produces

$$\sigma_{\theta_0}^2 = \frac{\sigma_0^2}{2r^2} \qquad (2.4\text{-}8)$$

The weight that each fictitious point should have is the reciprocal of its variance:

$$n_{c_0} = \frac{1}{\sigma_0^2} \tag{2.4-9}$$

Solving (2.4-8) for σ_o^2 and substituting into (2.4-9) produces

$$n_{c_0} = \frac{1}{2r^2\sigma_{\theta_0}^2} \tag{2.4-10}$$

When the two fictitious points are included in the summations in (2.4-2), their contribution to the outer summations (considering the means to be constant) are

$$\begin{aligned} \Delta s_{11} &= 2n_{c_0} r^2 \cos^2\theta_0 \\ \Delta s_{12} &= 2n_{c_0} r^2 \sin\theta_0 \cos\theta_0 \\ \Delta s_{22} &= 2n_{c_0} r^2 \sin^2\theta_0 \end{aligned} \tag{2.4-11}$$

Substituting (2.4-10) into (2.4-11) produces

$$\begin{aligned} \Delta s_{11} &= \frac{1}{\sigma_{\theta_0}^2} \cos^2\theta_0 \\ \Delta s_{12} &= \frac{1}{\sigma_{\theta_0}^2} \sin\theta_0 \cos\theta_0 \\ \Delta s_{22} &= \frac{1}{\sigma_{\theta_0}^2} \sin^2\theta_0 \end{aligned} \tag{2.4-12}$$

Note that r has canceled out of (2.4-12). However, in the inner summations in (2.4-2) (computing the means) it will not cancel out. Here r will appear in the first power in the denominator. Therefore, in the limit as r goes to infinity, these terms drop out and equation (2.4-12) correctly gives the entire effect of the fictitious points. It was considered above that σ_{θ_0} represented the accuracy of the direction of the a priori line if only the effect of the fictitious points were considered. However, again in the limit as r goes to infinity, the effect of the fictitious points interacting with the real points to affect the direction becomes zero, because the centroid of the fictitious points is constant (at the origin) and their weight becomes zero, according to ((2.4-10). Therefore, equation (2.4-12) correctly gives the effect of the a priori contrast knowledge.

The brightness and contrast adjustment can now be summarized as follows. For each position x_m, y_m, equation (2.4-2) is used, including the fictitious point for b_0 with weight given by (2.4-6), the results are augmented by (2.4-12) and are then used in (2.4-3), and the resulting value of s is used in (2.4-4) and (2.4-5).

Although the above computations correct for brightness bias and contrast changes between the pictures, explicit values for the changes (b and c) are not produced. These may be desirable, however. Values resulting from the application of the correlator to some parts of the pictures may be used to improve the estimates b_0 and c_0 given to the correlator when it is operating on other parts of the pictures. The above computations are done for all tentative matching positions within the search window, but single values of b and c are desired, computed from the apparently correct match.

Thus the adjusted values of b and c are computed in the following way. First, the matching point computed by (2.2-6) is rounded to the nearest pixel. Then the values of s_{11}, s_{12}, and s_{22} computed as described above for this x_m, y_m position (including the fictitious point and the Δ terms) are selected. The eigenvector of this distribution corresponding to the larger eigenvalue determines the scale factor c. (The eigenvector makes an angle θ with the a_1 axis, and $c = \tan\theta$, as previously described.) The direction of this eigenvector can be found from the following relationship:

$$c = \tan\theta = \frac{s_{22} - s_{11} + \sqrt{(s_{22}-s_{11})^2 + 4s_{12}^2}}{2s_{12}} \qquad (2.4\text{-}13)$$

Then b is the perpendicular distance from the origin to this eigenvector, where the line representing the eigenvector is assumed to pass through the mean of the distribution. Therefore,

$$b = \frac{\cos\theta \sum \frac{a_2}{\sigma_2^2} - \sin\theta \sum \frac{a_1}{\sigma_1^2}}{n} \qquad (2.4\text{-}14)$$

where the summations include the fictitious point for b_0, and n similarly includes n_{b_0}.

2.5 Interpolation

The computations described in the previous sections assume that the shift between the two pictures is always an integer number of pixels. In this section that assumption is removed, and the effects of noninteger values for x_c and y_c and how to deal with them are discussed. First will be discussed how to obtain satisfactory performance from the correlator in spite of these effects, without any particular attempt to produce subpixel accuracy in the x_2 and y_2 matching position estimates. Then a way of interpolating to produce this subpixel accuracy will be discussed briefly.

In cases where the correlation peak is broad (caused by a low signal-to-noise ratio), the smoothing process inherent in the moment computation for x_2, y_2, $\sigma^2_{x_2}$, $\sigma^2_{y_2}$, and $\sigma_{x_2y_2}$ cause a reasonable interpolation to be performed if the correct answer lies between pixels. However, when the correlation peak is sharp (caused by a high signal-to noise ratio), this will not happen, and the answer will tend towards the nearest pixel to the correct best match. This is not particularly serious insofar as it affects the position estimate, but it can have a serious effect on the variance estimate v_c and thus on the probability estimate also. This is because the E vector should be much smaller at the correctly interpolated point than it is at the nearest pixel, because of the sharp peak. Therefore, v_c may come out much too large, causing the probability estimate to be much too small, indicating a bad match, whereas the match actually is good but lies between pixels. To overcome this deficiency, the previously described computations are slightly modified.

Because of the tendency for x_2 and y_2 to tend towards the nearest pixel, the covariance matrix is augmented by adding $\frac{1}{12}$ to σ^2_x and σ^2_y. (Unity pixel spacing in x and y is assumed here, as before.) This is done because the variance of a uniform distribution with unity width is $\frac{1}{12}$. This in general overestimates the variance, since it assumes no inherent interpolation ability in the correlator.

The effect on the variance estimate will now be discussed. Without knowing something about the nature of the pictures, it is impossible to accurately correct for this phenomenon. However, a crude approximate correction can be made to v_c, and its weight n_c can be decreased by a liberal amount to allow for the uncertainty in this estimate.

Consider the various estimates of v which can be obtained from $\Sigma e_i^2/w_m^2$ or $2\sigma^2 s/w_m^2$, for each position within the search window. The minimum value of this quantity, previously denoted v_m, occurs at some particular position within the search window. Now consider the two such estimates for v obtained at one pixel displacements in the $\pm x$ direction from this minimum. Let Δv_x denote the difference between these two values. Similarly let Δv_y denote the difference between the two values displaced one pixel in the $\pm y$ directions. A sort of worst case assumption, which assumes that the true function wanted here is a V-shaped function in each dimension, leads to the conclusion that the true minimum of the function is less that v_m by $(|\Delta v_x|+|\Delta v_y|)/2$. However, if the corrected weighted-average value v_c from (2.3-10) is appreciably greater than v_m, the averaging process is doing some interpolation, and thus there is less need for a correction term. Therefore, the quantity $v_c - v_m$ is subtracted from this quantity. Furthermore, if v_c is greater than the v estimates at neighboring pixels, the corresponding values are replaced by v_c in the computation of Δv_x and Δv_y, for similar reasons. Since the resulting correction $(|\Delta v_x|+|\Delta v_y|)/2 - v_c + v_m$ represents sort of a worst case, it is divided by 2 to obtain an actual correction (not to be less than zero), which is subtracted from v_c, and the uncertainty in this correction is about the same magnitude. Thus, let

$$\Delta v_c = \max\left(0,\ \frac{1}{2}\left(\frac{|\Delta v_x| + |\Delta v_y|}{2} - v_c + v_m\right)\right) \tag{2.5-1}$$

The corrected value of v_c is then

$$\tilde{v}_c = \max(0,\ v_c - \Delta v_c) \tag{2.5-2}$$

The weight n_c must be adjusted to reflect the uncertainty in the correction. This can be done by using the fact that the mean of the chi-square distribution with n degrees of freedom is n and its variance about the mean is $2n$. Thus, if it is assumed that the additional uncertainty in $\tilde{v}_c$ caused by the uncertainty in this correction has a standard deviation equal to Δv_c, the corrected weight can be found from

$$\frac{1}{\tilde{n}_c} = \frac{1}{n_c} + \frac{\Delta v_c^2}{2\tilde{v}_c^2} \tag{2.5-3}$$

The corrected quantities $\tilde{v}_c$ and $\tilde{n}_c$ are used in place of v_c and n_c in the equations in the previous sections.

If the information in the pictures could actually be interpolated to produce interpixel values, not only could a better corrected variance and weight be produced than by using the above crude corrections, but the position estimates x_2 and y_2 could perhaps be refined to subpixel accuracy, with their variances becoming considerably less than the $\frac{1}{12}$ values used above. Ways of doing this will now be discussed.

In order to interpolate, some assumption must be made about the nature of the pictures. For example, if the pictures consisted of white noise, and they were sampled without any filtering to produce the digitized versions, there would be no way that any useful interpolation could be done (other than just setting all interpixel values to zero), because the interpixel values would be completely independent of the pixel values. At the other extreme, suppose that any content that the pictures contain at higher spatial frequencies than $\frac{1}{2}$ (the Nyquist frequency) in each dimension has been removed by filtering before the pictures are sampled. Then in the absence of noise it is possible to reconstruct the unsampled filtered pictures exactly by Fourier interpolation. Ordinarily the situation is between these extremes. There will be some content above the Nyquist frequency before sampling, and the sampling process folds this content into the frequencies below the Nyquist frequency. This process, known as "aliasing", contaminates these lower frequencies with this extraneous information, so that when interpolation is done based upon the information in these lower frequencies, errors are produced. In order to know how to interpolate the data to keep these errors small, some statistical knowledge about the amount of aliased content must be available.

If a certain power spectrum of true picture content could be assumed, the amount of aliasing could be computed at each frequency, and thus a weighting function of frequency could be derived, based on the accuracy of each spatial frequency as deduced from its amount of contamination. Then a Fourier interpolation could be done, but instead of cutting off precisely at the the Nyquist frequency, the computed frequency components would be multiplied by the weighting function, causing a gradual cutoff as the Nyquist frequency is passed. The result could then be transformed back to the space domain to obtain interpolated data. Also, an additional variance component representing the uncertainty in the interpolation would be computed, as a function of the interpolation position. (The variance due to noise usually is less for the interpolated points because of the averaging that occurs in the interpolating process, but the additional variance caused by the aliased picture content usually causes the total variance to increase.)

An interpolating version of the correlator has been produced based on the above reasoning. It interpolates a_2 to a finer sampling interval in both dimensions, with the option of two different assumptions about the nature of the power spectrum, one of which produces linear interpolation. Then a_1 is compared to the interpolated a_2 at the original sampling interval as required for (2.2-4) or (2.4-2), with σ_2^2 augmented for the interpolating error, but this is done for every position within the search window at the interpolated sampling interval to produce the summations in (2.2-6) and (2.2-7). Then the approximate interpolation corrections described above are applied at this interpolated sampling interval instead of at the original sampling interval. (If the interpolation is done at a fine enough sampling interval, this last step is not necessary, but it is usually not known beforehand how fine an interval is needed.)

Although this interpolating version of the correlator can produce greater accuracy in some cases, it is very slow. It was developed for a special application while the author was at Lockheed, and it has not been used in any of the other research described in this thesis. Therefore, it will not be described in further detail here. (The usual version of the correlator does use the approximate interpolation corrections previously described, however.)

2.6 Color

The description of the correlator in the previous sections assumes that the pictures are monochromatic, and this is the case in the implemented version of the correlator. However, most of these computations generalize readily to handle color pictures.

If color pictures are used, then for each pixel the scalar a_1 or a_2 is replaced by the vector A_1 or A_2, with one component for each primary color being used. Of course, it is not necessary to use three primary colors, as with human vision. In designing a vision system to perform a particular task, the number of primary colors and the bands of wavelengths to which they correspond would be chosen according to how the content of typical scenes varies at different wavelengths. Thus the A_1 and A_2 vectors might well have more than three components. (It is not even necessary that all of these components be obtained from brightness at certain wavelengths of light. Some might come from other information, such as parameters describing texture obtained at extra high resolution, or sonar data.)

In equation (2.2-4), where the square of a difference e_i^2 is used with monochromatic pictures, the sum of the squares of the components of the difference of the two vectors could be used instead for color pictures, if all components of the vector were equally accurate. However, in general the noise in each primary color will be different, and thus the square of the difference of each component must be divided by the variance of that component individually, and these results summed to replace $e_i^2/(\sigma_1^2+\sigma_2^2)$ in (2.2-4). A more general form to use when the noise in the different primary colors is correlated would be to use the quadratic form produced from the vector and the inverse of the covariance matrix which describes the noise. Equation (2.2-4) would then be replaced by

$$w = \exp\left(-\tfrac{1}{2} \sum (A_1-A_2)^{\mathrm{T}}(S_1+S_2)^{-1}(A_1-A_2)\right) \tag{2.6-1}$$

where S_1 and S_2 are the covariance matrices of the vectors A_1 and A_2, these quantities are to be aligned according to the current position within the match window as described in Section 2.2, and the summation is over all positions in the match window. (The subscript i has now been dropped, and the dependence upon position within the match window is now implicit.) However, ordinarily the noise in the different channels is uncorrelated, and thus S_1 and S_2 are diagonal matrices, and (2.6-1) reduces to the simpler form described above.

In the variance estimation, there will now be a separate component of variance to estimate for each primary color. Equations (2.3-2), (2.3-10), (2.3-11), (2.3-12), and (2.3-13) can be applied independently for each component. However, a more general form is possible, as with (2.6-1), in which a complete covariance matrix is estimated. To do this, instead of squaring each single component in (2.3-2) and (2.3-11), the outer product of a vector with itself is taken (that is, the matrix product of the vector times the transpose of the vector) to produce a square matrix. As stated above, this would seldom be necessary.

In order to obtain the probability estimate, a separate F test could be computed for each primary color. The product of the resulting probabilities could be used as the result.

If brightness bias and contrast change are to be adjusted, there are several possibilities, depending on exactly what is wanted. If a separate adjustment for each primary color is wanted, the equations in Section 2.4 can be used separately on each component. However, if a single adjustment affecting all channels equally is desired, these computations would have to modified slightly to include this constraint. The most general linear relationship between the two vectors A_1 and A_2 would be to premultiply one of them by a square matrix of contrast coefficients and then to add a vector of bias coefficients. How all of these coefficients could be determined is beyond the scope of this thesis, and it is hard to imagine how such a relationship (including nonzero off-diagonal elements in the matrix) would be produced by a reasonable vision device.

2.7 Speed

In processing a typical stereo pair of pictures by the technique to be described in Chapter 4, the correlator is applied thousands of times. More than half of the total computing time through all of the computations described in this thesis can be spent in the correlator. Therefore, considerable effort was put into making the correlator efficient.

It might appear that most of the time in the correlator would be spent in computing the sum of squares of differences needed in (2.2-4) or the sum of products needed in (2.4-2). (For example, if $w_m = 8$ and $w_s = 8$, there would be 8^4=4096 total terms in all of the summations needed in (2.2-4) over the search window.) If the code for these computations were written in a straightforward way using nested FOR loops, this would be the case. (Fourier-transform methods are faster only with large windows.) However, a special method for computing the needed sum of squares of differences developed by Moravec [1977] is used in the implemented correlator. This utilizes the fact that the pixel brightness values can be represented by small integers. It does the difference by indexing with a register and does the squaring by a table lookup. The machine code for this is compiled in line for the entire match window by the program. Then the main program uses this code for each position within the search window. The entire inner loop of this code (each term of the summation) consists of one Move Negative instruction and one fixed-point Add instruction and requires about one microsecond on the PDP KL10.

When the bias and contrast adjustment is done, a sum of products is needed in (2.4-2). Since $2a_1a_2 = a_1^2+a_2^2-(a_1-a_2)^2$, this is computed from the sum of squares of differences and two sum of squares. The sum of a_1^2 over the match window is constant, and the sum of a_2^2 over the match window is computed quickly for each position within the search window by the usual moving-average technique of adding new points and subtracting old points as the match window moves. Alternatively, this latter sum of squares could be computed by specially compiled code similar to that for the sum of squares of differences.

Various other techniques are used to speed up the computations. For example, the exponential function needed in (2.2-4) is computed by a table lookup, and, if the argument is so large that w will be negligibly small, the computations using w are bypassed for this position within the search window. Also, in computing the moments according to (2.2-6) and (2.2-7), symmetry is utilized, so that the number of multiplications is cut almost by a factor of four.

The actual amount of time used by the correlator depends upon the sizes of the search window and match window, the signal-to-noise ratio, whether brightness bias and contrast are adjusted, whether the variance upper limit from high frequencies is used, and the accuracy of the a priori variance estimate. If $w_s = 8$ and $w_m = 8$, the CPU time on the PDP KL10 ranges from about 14 milliseconds to about 70 milliseconds.

Chapter 3

STEREO CAMERA CALIBRATION

In order to do the computations to be described in Chapters 4 and 5, the stereo camera model (the relative position and orientation of the two cameras which produced the stereo views) must be known. In some cases this is known beforehand. If it is not, the information to calibrate the stereo camera model can come from the pictures themselves. This chapter describes such a self-calibration method. This problem is known as the relative orientation problem in photogrammetry. (For example, see Schut [1957 and 1959].) However, the data for the adjustment is obtained in a different way here, as described in Section 3.1, the additional features described in Section 3.2 are used, and the basic formulation of the problem here is somewhat different from the usual approaches in photogrammetry, as pointed out in Section 3.3. (Some of these solutions used in photogrammetry allow the use of many cameras, instead of just two, however, as in Davis [1967].) An equivalent problem occurs when a single fixed camera views a moving object at different times. Such a problem is treated by Ullman [1976], although he considers mainly the case of three views of only four points. Here, only two views are used, and thus at least five points are required for a solution if there is no other information.

3.1 Points for Self-Calibration

In order to extract the necessary information from the pictures themselves, some features or points must be matched between the two pictures. Since the camera model is not yet known, this requires a two-dimensional search. However, the number of matches required is not large. It should be at least as great as the number of camera model parameters being adjusted, and preferably considerably greater in order to improve the accuracy and to detect errors. From 20 to 50 matches scattered over the picture is reasonable.

The implemented program uses Moravec's interest operator and binary-search correlator to perform this matching (Moravec [1977 and 1980]). First, the interest operator is applied to Picture 1 and finds small features with high information content. It discriminates against features with low contrast or with primarily one-dimensional information. Then the binary-search correlator finds the corresponding points in Picture 2. It uses a coarse-to-fine method, starting with the whole picture and rapidly homing in on the matching point. However, some of the matches that it makes are incorrect.

These matched points are then refined by the correlator described in Chapter 2. There are three reasons for this step. The positional accuracy of a match may be improved, its accuracy can be estimated in order to provide appropriate weight in the camera model adjustment, and the probability value computed by the correlator can be used to reject some of the incorrect matches. Actually, since the interest operator detects only high-contrast features, the accuracy of the matches is usually very good (limited primarily by the pixel spacing). Thus usually there is not much improvement to be made in their accuracy (unless the interpolating version of the correlator is used), and the standard deviations are usually near the minimum of $1/\sqrt{12}$ of the pixel spacing. Therefore, the first two reasons are not important for most points. However, the third reason (rejection of bad points) is quite useful. The implemented program rejects any point with a probability less than 0.1. This rejects a few good points (about 10% of them if the probability value is correct), and it still lets a few bad points through, but by reducing the number of bad points it helps the camera model adjustment significantly, both in speed and in likelihood of success.

Finally, the image-plane coordinates of the points are corrected for camera distortion to make them equivalent to those produced by a central projection.

The result for each remaining matched point consists of the image coordinates x_1 and y_1 for the point in Picture 1, the image coordinates x_2 and y_2 for the point in Picture 2, and the variances σ_x^2 and σ_y^2 and the covariance σ_{xy} of the image coordinates of the point in Picture 2. (The subscript "2" is dropped from the subscripts of "σ" in order to avoid confusion with other subscripts to come. This should cause no ambiguity, since x_1 and y_1 are considered to be known exactly and thus have no variances and covariances associated with them.)

3.2 Additional Error

The errors indicated by the accuracy estimates produced by the correlator are presumably independent for each point. However, there may be other sources of error which the correlator cannot estimate, and some of these may be correlated between different points. For example, there may be some residual distortion in the pictures that has not been corrected. If this is different in the two pictures or if the pair of matching points are in different portions of the two pictures where this residual distortion is different, an error is produced. For different point pairs in widely different positions in the pictures, the residual distortion may be quite different, but, since distortion usually varies slowly across a picture, the effect on nearby points may be quite similar. These additional errors and their correlations, if any, must be taken into account in order to produce the correct weights for the camera model adjustment and the correct error propagation into the results.

One way to obtain the necessary information is from the distortion correction measurement. By analyzing the residuals of the distortion adjustment, the magnitude of the errors and how rapidly they vary across the picture can be estimated. Experience with previous distortion measurements and the variation of distortion with time on the camera or cameras of the same type may be helpful. Also, the camera model adjustment itself can estimate the variance of the additional error (but not the precise way in which it varies across the picture).

Ideally, what we would like to have is a complete covariance matrix for this additional error for all of the points. In practice, such complete information usually is not available. However, if the magnitude of the error is known and the approximate image-plane distance over which it is highly correlated is known (called here the "correlation distance"), a reasonable approximation is possible. The implemented program assumes that the correlation coefficient of the additional error is a Gaussian function of the distance between the points in the image plane. Thus the covariance of the additional error for points i and j is assumed to be $\gamma \exp(-d_{ij}^2/2c^2)$, where γ is the variance of the additional error, c is the correlation distance, and d_{ij} is the distance between the two points. (This distance may be different in the two pictures. The average of the two results can be used.) It is assumed here that the additional error is uncorrelated between x and y and has the same variance in x and y. Thus the elements of the covariance matrix for total error (denoted by a tilde) are assumed to be as follows:

$$
\begin{aligned}
\tilde{\sigma}_{x_i}^2 &= \sigma_{x_i}^2 + \gamma \\
\tilde{\sigma}_{y_i}^2 &= \sigma_{y_i}^2 + \gamma \\
\tilde{\sigma}_{x_i y_i} &= \sigma_{x_i y_i} \\
\tilde{\sigma}_{x_i x_j} &= \gamma \exp\left(-\frac{d_{ij}^2}{2c^2}\right), \text{ if } i \neq j \\
\tilde{\sigma}_{y_i y_j} &= \gamma \exp\left(-\frac{d_{ij}^2}{2c^2}\right), \text{ if } i \neq j \\
\tilde{\sigma}_{x_i y_j} &= 0, \text{ if } i \neq j
\end{aligned}
\qquad (3.2\text{-}1)
$$

where i and j denote any two points.

The correlation distance c is considered to be a given quantity. However, the additional variance γ can be adjusted by the program according to the method described in Appendix A.

The covariance matrix, whether obtained from precise knowledge of the individual correlations or from (3.2-1), could be used as S_U in Appendix A, which is inverted to obtain the weight matrix W to be used in (A.1-17). However, in order to save computation time, the implemented program uses the solution according to (A.1-23), partitioned into the separate points. Strictly speaking, this would require that the correlations between different points be zero. However, because of the fact that the effect of a point on the solution varies slowly as the point is moved across the picture, and because of the fact that with the approximation of (3.2-1) the correlations are negligible for far-apart points and the additional variance is equal for all points, the approximation in Section A.2 can be and is used in the implemented program, and this permits the solution to be partitioned by points. The specially augmented variances and covariances produced by this approximation will be used to obtain the weights for the solution.

3.3 Criterion for Adjustment

This section will describe how the image plane measurements described in the previous two sections are used to obtain the discrepancies and their weights so that the adjustment for the camera model parameters can be performed. (The adjustment will be performed according to the method described in Appendix A, which is basically a weighted least-squares adjustment with some modifications. Its particular form for this problem will be outlined in the next section.)

There are many possible ways of formulating the problem, according to what are defined to be the discrepancies whose weighted sum of squares is to be minimized. Some of the other methods that have been used are discussed by Schut [1957 and 1959]. The method used here defines the discrepancies as distances in the image plane, closely related to the actual measured quantities. All of these methods are approximately equivalent as long as the appropriate partial derivatives relating the observations to the discrepancies are included in the formulation, according to the method described by Brown [1955 and 1957], and as long as no points appear to be beyond an infinite distance. (A more readily accessible description can be found in Mikhail [1976].) However, the method used here avoids the need to do this, and permits the use of the simpler formulation described in Appendix A. It also permits the use of points that appear to be beyond infinity. This is important in some cases, because observation errors may cause a distant point to appear to be beyond infinity. (The complete method, described in the next section, also includes the features of variance adjustment and automatic editing.)

As formulated in Appendix A, the general solution method requires measurements to be made directly on quantities that are functions of the parameters. However, this is not quite the situation that we have. Here the directly observable quantities are x_1, y_1, x_2, and y_2. The method used by Brown mentioned in the previous paragraph can handle such situations within the general formulation. However, this is not necessary for our purposes here. We will merely propagate the error estimates of the actual observations into the quantity that we use as the discrepancy, in order to obtain the correct weights, and will consider the observations to be measurements directly on the discrepancy on any one iteration. Since the discrepancy that we will use will be some distance in the Camera 2 film plane, and since we will consider the measurements to be made in this plane, the transformation between them is linear and thus this error propagation will be exact, although the fact that the propagation depends upon the camera model parameters that are being adjusted causes a slight departure from optimality.

Consider the point x_1, y_1 in the Camera 1 image plane projected as a ray in space from the Camera 1 center of projection and then consider this ray projected back into the Camera 2 image plane. The result is a line segment, because a point at an infinite distance on this ray projects into a specific point in the Camera 2 film plane (unless the ray is parallel to the film plane). The coordinates of this infinity point (in the Camera 2 film plane) which defines the end of the line segment are denoted by x_0, y_0. The direction of the line segment (away from the infinity point) is given by the direction cosines c_x and c_y relative to the x and y axes, respectively. (These quantities x_0, y_0, c_x, and c_y and their partial derivatives relative to the camera model parameters are computed from x_1, y_1, and the stereo camera model. The details of this computation for the camera model formulation used in the current work are given in Appendix B.)

The discrepancy e consists of a component of the distance from the measured point x_2, y_2 in the Camera 2 image plane to the nearest point of the line segment defined by x_0, y_0, c_x, and c_y. If the point x_2, y_2 is beyond the infinity point x_0, y_0, there are two components of the distance between these two points, and thus there are two observations for this point (two components of the vector E). Otherwise, e consists of the perpendicular distance from x_2, y_2 to the line, and there is only one observation for this point.

It remains to define precisely what is meant by "beyond the infinity point." If the perpendicular projection onto the line were used, the projected point would be considered to be beyond the end of the line segment if $(x_2 - x_0)c_x + (y_2 - y_0)c_y < 0$. However, because of the nature of the errors in x_2 and y_2, a different projection should be used. If a normal distribution of errors is assumed, the correct projected point is the tangent point of the line to an error ellipse about x_2, y_2. This will be discussed further in Chapter 5. Using (5.1-5), substituting (5.1-3), ignoring the denominator (since it is always positive and only the sign of the resulting quantity needs to be considered here), and recognizing that total error (as from (3.2-1)) should be used here produces the quantity $(c_x\tilde{\sigma}_y^2 - c_y\tilde{\sigma}_{xy})(x_2 - x_0) + (c_y\tilde{\sigma}_x^2 - c_x\tilde{\sigma}_{xy})(y_2 - y_0)$.

If $(c_x\tilde{\sigma}_y^2 - c_y\tilde{\sigma}_{xy})(x_2 - x_0) + (c_y\tilde{\sigma}_x^2 - c_x\tilde{\sigma}_{xy})(y_2 - y_0) \geq 0$, then the point x_2, y_2 as projected according to the above description does not lie beyond the the end of the line segment defined by x_0, y_0, c_x, and c_y, and the discrepancy e is the perpendicular distance from the point to the line. Therefore,

$$
\begin{aligned}
e &= (y_2 - y_0)c_x - (x_2 - x_0)c_y \\
\frac{\partial e}{\partial g} &= (y_2 - y_0)\frac{\partial c_x}{\partial g} - (x_2 - x_0)\frac{\partial c_y}{\partial g} - c_x\frac{\partial y_0}{\partial g} + c_y\frac{\partial x_0}{\partial g}
\end{aligned}
\tag{3.3-1}
$$

where g represents any of the camera model parameters. (The way in which the polarity of

e is defined does not matter, as long as the polarity of its derivatives is consistent with this. When the partial derivatives from (3.3-1) are assembled into the P matrix needed for the solution described in Appendix A, their signs will be changed, because P is defined in Section A.1 as the partial derivatives of F, which appears in the equation for E with a minus sign.) The variance of e representing errors in x_2 and y_2 is then

$$\sigma_e^2 = c_x^2\sigma_y^2 - 2c_x c_y \sigma_{xy} + c_y^2\sigma_x^2 \tag{3.3-2}$$

This equation holds whether the σ's represent only the estimates from the correlator, total error according to (3.2-1) (in which case the symbols should be $\tilde{\sigma}$), or augmented error according to (A.2-1) (in which case the symbols should be $\hat{\sigma}$). However, since the additional term for total error or augmented error will be the same for both σ_x^2 and σ_y^2 and zero for σ_{xy}, and since $c_x^2+c_y^2 = 1$, the additional variance or augmentation variance can just be added to σ_e^2 from (3.3-2). The reciprocal of the resulting value for augmented variance will be used for the weight of this point in the adjustment. (This will be made explicit in the next section.)

On the other hand, if $(c_x\tilde{\sigma}_y^2-c_y\tilde{\sigma}_{xy})(x_2-x_0) + (c_y\tilde{\sigma}_x^2-c_x\tilde{\sigma}_{xy})(y_2-y_0) < 0$, there are discrepancies (the two components of the vector E) which are the two components of the distance from the point x_2, y_2 to the end of the line segment (x_0, y_0). Any two orthogonal components can be used here; for convenience we will use the x and y components. Therefore,

$$E = \begin{bmatrix} x_2 - x_0 \\ y_2 - y_0 \end{bmatrix}$$

$$\frac{\partial E}{\partial g} = \begin{bmatrix} -\dfrac{\partial x_0}{\partial g} \\ -\dfrac{\partial y_0}{\partial g} \end{bmatrix} \tag{3.3-3}$$

The covariance matrix of E is the same as the covariance matrix of x_2 and y_2, for any of the three types of error in x_2 and y_2. The weight matrix for this point is the inverse of the augmented covariance matrix. (Notice that in this case the problem has reduced to the usual problem discussed in Appendix A. When the signs of the partial derivatives are changed as mentioned above for insertion into the P matrix, the P matrix is seen to contain the partial derivatives of x_0 and y_0 with respect to the camera model parameters. Thus the observations in effect are directly on the quantities x_0 and y_0. This is exactly the actual situation, since x_2 and x_2 can be considered to be observations on x_0 and y_0, respectively, when the point appears to be beyond infinity.)

3.4 Summary of Adjustment

The previous sections in this chapter have described how the data for the stereo camera model adjustment are obtained and the particular way in which they are used in the adjustment. Appendix A describes the general methods that are used in the adjustment. Appendix B describes the way in which various quantities used in the adjustment are related to the camera model parameters. This section gives a summary of the stereo camera model adjustment, showing how these pieces tie together and filling in a few details, more or less as it is currently implemented.

The given quantities are as follows: a priori values of the five camera model parameters azimuth, elevation, pan, tilt, and roll described in Appendix B, denoted by the vector G_0; the covariance matrix of these a priori values, denoted by S_{G_0}; the principal distances of the two cameras, denoted by f_1 and f_2, described in Appendix B (considered to be known exactly); the a priori value of the additional variance, denoted by γ_0; the standard deviation of γ_0, denoted by $\sigma^2_{\gamma_0}$; the correlation distance of the additional error, denoted by c; the maximum number of points to edit, denoted by n_e; and for each matched point the values x_1, y_1, x_2, y_2, σ^2_x, σ^2_y, and σ_{xy} determined by the correlator, as described in Section 3.1. The following quantities are to be computed: adjusted values of the five camera model parameters, denoted by the vector G; the covariance matrix of these adjusted values, denoted by S_G; and the adjusted value of the additional variance, denoted by γ. The steps in the computation are as follows.

1. (Begin edit loop.) Correction factors for correlated errors are computed for each point i as follows:

$$k_i = \sum_j \exp\left(-\frac{d^2_{ij}}{2c^2}\right)$$

$$\kappa_i = \sum_j \left[\exp\left(-\frac{d^2_{ij}}{2c^2}\right)\right]^2$$

derived from (3.2-1), (A.2-1), and the approximation following (A.3-4). (The quantity k_i is the sum of the correlations of the additional error between this point and every point and will be used in obtaining weights for the main adjustment. The quantity κ_i is the sum of the squares of these correlations and will be used in obtaining weights for the variance adjustment.) After the first time through this step, the above summations are updated by simply subtracting the terms for the rejected points. If $c = 0$, then $k_i = 1$ and $\kappa_i = 1$ for all points.

2. As initial approximations, G is set equal to G_0, and γ is set equal to γ_0. Initially, $r_i = 1$ for all points.

3. (Begin inner iteration loop.) The a priori values of the parameters are used as follows:

$$H_0 = S^{-1}_{G_0}$$

$$C_0 = H_0(G_0 - G)$$

4. For each point the quantities x_0, y_0, c_x, and c_y and their partial derivatives are computed as described in Appendix B. (The subscript i is omitted from these and from the input values x_1, y_1, x_2, y_2, σ^2_x, σ^2_y, and σ_{xy} for each point for simplicity, but all other quantities that depend on the point carry this subscript.) Then, if $[c_x(\sigma^2_y+\gamma)-c_y\sigma_{xy}](x_2-x_0) + [c_y(\sigma^2_x+\gamma)-c_x\sigma_{xy}](y_2-y_0) \geq 0$ the following is done for this point:

$$n_i = 1$$

$$e_i = (y_2 - y_0)c_x - (x_2 - x_0)c_y$$

$$P_i = -\frac{\partial e_i}{\partial G} = -(y_2 - y_0)\frac{\partial c_x}{\partial G} + (x_2 - x_0)\frac{\partial c_y}{\partial G} + c_x\frac{\partial y_0}{\partial G} - c_y\frac{\partial x_0}{\partial G}$$

$$\sigma^2_{e_i} = c^2_x\sigma^2_y - 2c_xc_y\sigma_{xy} + c^2_y\sigma^2_x$$

$$w_i = \frac{1}{\sigma^2_{e_i} + k_i\gamma}$$

$$C_i = P^T_iw_ie_i$$

$$H_i = P^T_iw_iP_i$$

$$\omega_i = \frac{1}{2}\cdot\frac{1}{(\sigma^2_{e_i}+\gamma)^2 + (\kappa_i-1)\gamma^2} = \frac{1}{2(\sigma^4_{e_i} + 2\gamma\sigma^2_{e_i} + \kappa_i\gamma^2)}$$

$$\chi_i = \omega_ir_ie^2_i$$

$$\upsilon_i = \omega_i\sigma^2_{e_i}$$

On the other hand, if $[c_x(\sigma^2_y+\gamma)-c_y\sigma_{xy}](x_2-x_0) + [c_y(\sigma^2_x+\gamma)-c_x\sigma_{xy}](y_2-y_0) < 0$ the following is done for this point:

$$n_i = 2$$

$$E_i = \begin{bmatrix} x_2 - x_o \\ y_2 - y_o \end{bmatrix}$$

$$P_i = -\frac{\partial E_i}{\partial G} = \begin{bmatrix} \frac{\partial x_o}{\partial G} \\ \frac{\partial y_o}{\partial G} \end{bmatrix}$$

$$W_i = \begin{bmatrix} \sigma_x^2 + k_i\gamma & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 + k_i\gamma \end{bmatrix}^{-1}$$

$$C_i = P_i^T W_i E_i$$

$$H = P_i^T W_i P_i$$

$$\Omega_i = \frac{1}{2} \begin{bmatrix} \sigma_x^4 + 2\gamma\sigma_x^2 + \kappa_i\gamma^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_y^4 + 2\gamma\sigma_y^2 + \kappa_i\gamma^2 \end{bmatrix}^{-1}$$

$$\chi_i = [\,1 \quad 1\,]\,\Omega_i \begin{bmatrix} r_{x,i}(x_2 - x_o)^2 \\ r_{y,i}(y_2 - y_o)^2 \end{bmatrix} = (\omega_{11,i} + \omega_{12,i}) r_{x,i}(x_2 - x_o)^2 + (\omega_{12,i} + \omega_{22,i}) r_{y,i}(y_2 - y_o)^2$$

$$\upsilon_i = [\,1 \quad 1\,]\,\Omega_i \begin{bmatrix} \sigma_x^2 \\ \sigma_y^2 \end{bmatrix} = (\omega_{11,i} + \omega_{12,i})\sigma_x^2 + (\omega_{12,i} + \omega_{22,i})\sigma_y^2$$

$$\omega_i = [\,1 \quad 1\,]\,\Omega_i \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \omega_{11,i} + 2\omega_{12,i} + \omega_{22,i}$$

However, if n_i for this point changed from the previous iteration, $\chi_i = \upsilon_i = \omega_i = 0$.

5. The quantities C, H, χ, υ, and ω are computed by summing over all points (currently being used) the corresponding quantities with the subscript i, as computed in step 4. For C and H, C_0 and H_0 are included in the summations.

6. Then

$$S_C = H^{-1}$$

$$D = H^{-1}C$$

(from A.1-23)), but the elements of D are limited so that their absolute value does not exceed 0.5 radians (thus limiting wild excursions that may take place on early iterations if the initial approximation is poor).

7. For each point the following is done. If $n_i = 1$,

$$\sigma^2_{v_i} = \sigma^2_{e_i} + \gamma - \min\left(P_i S_C P_i^T,\ \gamma + \frac{\sigma^2_{e_i} P_i S_C P_i^T}{\sigma^2_{e_i} + \kappa_i \gamma}\right)$$

$$r_i = \frac{\sigma^2_{e_i} + \gamma}{\sigma^2_{v_i}}$$

(from (A.1-22), (A.2-2), and (A.3-2).) If $n_i = 2$, the same thing is done to compute $r_{x,i}$ and $r_{y,i}$, except that for the former the first element of E_i is used instead of e_i, $\sigma^2_{x_i}$ is used instead of $\sigma^2_{e_i}$, and the first row of P_i is used for P_i, and for the latter the second element of E_i is used instead of e_i, $\sigma^2_{y_i}$ is used instead of $\sigma^2_{e_i}$, and the second row of P_i is used for P_i.

8. If the solution has started to converge (indicated by the maximum absolute value of an element of D being less on some iteration than on the previous iteration), the variance adjustment is done as follows:

$$\gamma_c = \frac{X - U}{\omega}$$

$$\sigma^2_{\gamma_c} = \frac{1}{\omega}$$

$$\gamma = \frac{X - U + \dfrac{\gamma_0}{\sigma^2_{\gamma_0}}}{\omega + \dfrac{1}{\sigma^2_{\gamma_0}}} = \frac{\dfrac{\gamma_c}{\sigma^2_{\gamma_c}} + \dfrac{\gamma_0}{\sigma^2_{\gamma_0}}}{\dfrac{1}{\sigma^2_{\gamma_c}} + \dfrac{1}{\sigma^2_{\gamma_0}}}$$

(from (A.3-5)), but γ is not to be less than zero.

9. If the solution has started to converge (as indicated in step 7), the convergence

acceleration procedure described in Section A.4 is applied to G and D. For this purpose, the variance γ is considered to be a sixth parameter (sixth element of G), scaled by dividing it by f_2^2 (with its difference from the preceding iteration being a sixth element of D.) However, if for any point the condition of being beyond infinity changes from the previous iteration (that is, n_i changes), the acceleration procedure is restarted. Whether D has been changed by the acceleration procedure or not, it is added to G from the previous iteration to produce the new G.

10. If the greatest absolute value of an element of D (before acceleration) is is less than 10^{-6} radian, and the change in γ from the previous iteration is less than $10^{-4}(\gamma + U/\omega)$, then go to step 11 (exit from the inner iteration loop). Otherwise, if too many iterations have occurred, give up. Otherwise, go to step 3. (End inner iteration loop.)

11. If $n_e = 0$, finish successfully. Otherwise, if there is no tentatively rejected point (this is the first time through the edit loop), go to step 16.

12. For the last point tentatively rejected, the quadratic form of its residuals with the inverse of their covariance matrix is computed according to (A.5-2). (If $n_i = 1$, this reduces to the ratio of the square of its residual to the variance of the residual, where the variance of the residual is computed as in step 7 but with a plus sign instead of the minus sign.) If the result is greater than 9 if $n_i = 1$ or 16 if $n_i = 2$, go to step 15.

13. If the total number of rejected points equals n_e, go to step 17 (exit from the edit loop).

14. The following F test is computed:

$$p_F\left(\frac{\gamma_c+a}{\gamma_0+a},\ \frac{2(\gamma+a)^2}{\sigma^2_{\gamma_c}},\ \frac{2(\gamma+a)^2}{\sigma^2_{\gamma_0}}\right) \overset{?}{>} 0.02$$

where $a = U/\omega$, and where $p_F(f, n_1, n_2)$ is the probability that the ratio of a chi-square estimate of a variance with n_1 degrees of freedom to another chi-square estimate of the same variance with n_2 degrees of freedom will exceed f. (The above use of this test is a crude approximation based upon the assumption that the sum of the additional variance and the input point variance weighted over all points with weight ω_i has the chi-square distribution.) If this test passes (the above inequality is true), go to step 17 (exit from the edit loop). Otherwise, go to step 16.

15. Reject all of the tentatively rejected points. If the total number of rejected points equals n_e, give up.

16. For each current point, the quadratic form of its residuals with the inverse of their covariance matrix is computed according to (A.5-1). (If $n_i = 1$, this reduces to the

ratio of the square of its residual to the variance of the residual, where the variance of the residual is computed as in step 7.) This quadratic form is divided by 9 if $n_i = 1$ or by 16 if $n_i = 2$. The current point with the largest resulting value is tentatively rejected. Then go to step 1. (End edit loop.)

17. All tentatively rejected points are reinstated, the solution backs up to the one computed using these points, and the problem is finished successfully.

Both intuition from the nature of the problem and experience with the program indicate that there are no other local minima of the total quadratic form of the solution near the absolute minimum. Therefore, if the initial approximation is near the correct solution, the solution should converge to it, if it converges at all. However, there is another minimum in cases where one camera is roughly in front of or in back of the other (azimuth ≈ 0 or π). This local minimum occurs when the front-back positions of the cameras are reversed, for then most of the points appear to be beyond infinity. Tests could be put into the program to detect this condition and to change to the other solution, but this has not been done. If the initial approximation is reasonable, there should be no problem with this phenomenon.

As mentioned in Section A.1, if the iterative solution converges to the absolute minimum, it produces the exact weighted least-squares solution. Other properties, such as the estimates of accuracy of the adjusted parameters, are approximately correct as long as the problem is approximately linear. This is the case as long as no points are near infinity. However, if a point x_2, y_2 is near the infinity point x_0, y_0 (compared to its standard deviation), a large nonlinearity is introduced. This will cause, among other things, the error estimates represented by S_G to be underestimates if the point x_2, y_2 lies beyond the infinity point or overestimates if the point appears to be closer than infinity. Furthermore, this nonlinearity is caused by a discontinuity. Thus using the second derivatives (as described in Section A.1) probably would not help. That is, equation (A.1-19) may be no better than (A.1-20) in this case. This effect also affects the convergence, especially of Newton's method, which uses the second derivatives. This is the reason for restarting the acceleration convergence procedure when a point moves across an infinity point.

Chapter 4

STEREO MATCHING

This chapter describes a method of matching points densely over an entire scene, so that when the distances to these points are computed as described in the next chapter a dense depth map will be produced.

In Chapter 2 a stereo correlator was discussed which can refine a local tentative match between a stereo pair of pictures. However, it is necessary to have a means of deciding where to apply the correlator and to have a decision criterion for deciding which matches to accept. This could be done independently for each point to be matched. Another possibility is to use continuity constraints to force a smooth surface to be produced, with only a very local search used. Some form of region growing as in Hannah [1974] might be used in the latter case. The approach adopted here lies between these two extremes. The stereo disparities are allowed to vary in an arbitrary way over the picture, subject to some mild local continuity constraints discussed later, which eliminate some incorrect matches that otherwise would be made. Furthermore, by first trying a match with approximately the same stereo disparity as neighboring points that already have been matched, the search can be eliminated for many points. The acceptance of matches is guided by the probability values returned by the correlator and by agreement with neighboring matches. No claim is made that this approach is optimum for any particular type of scene, but it seems to work well for the type of scene considered in this research (outdoor scenes with various objects strewn about). (The method of Levine *et al.* [1973] has some features that perhaps should be included in an operational system, such as the use of an adaptive correlation window size.)

Because the stereo camera model is known at this point, the search that needs to be performed is only one-dimensional. A point in Picture 1 corresponds to a ray in space, which, when projected into Picture 2, becomes a line segment terminating at the point corresponding to an infinite distance along the ray. Therefore, a search along this line segment suffices.

Because the approach used here is based upon area correlation, the first step in the matching process is to divide the master picture (here called picture 1) into small areas equal in size to the match window of the correlator, for each of which a matching area in the other picture (picture 2) is desired. (It usually is desirable to have these windows to be adjacent and nonoverlapping. Since the correlator match window is square, this results in a square tesselation of Picture 1.) These areas must be selected in some order to be matched.

One possibility for ordering areas to be matched would be to start with the points which were produced by the interest operator and binary-search correlator and were not rejected by the camera model solver and then to work outwards in all directions from these points, since these points are very likely to be correctly matched. However, there may be regions separated from all of these points by disparity discontinuities or unmatchable regions, so this method does not alleviate the search procedure from the necessity to be self-starting. (A similar method that may produce denser starting points is the tie-point method of Levine, *et al.* [1973].) The method that actually is used simply starts with a column at the left edge of the picture and works to the right a column of areas at a time. In addition to simplicity, this method has the following advantage when, as is customary, Camera 2 is to the right of Camera 1. The line segment in Picture 2 corresponding to a point in Picture 1 then is directed to the left from the infinity point. Thus, once a few columns in Picture 1 have been matched, if a search is made starting at the infinity point and proceeding leftward, eventually areas will be encountered in Picture 2 that already have been matched. If it can be assumed that there are no foreground objects that can be seen around so that the left camera sees some background points to the left of the object and the right camera sees these same points to the right of the object, then once a sufficient number (unlikely to be incorrect matches) of previously matched areas have been encountered in this manner, the search can be terminated, as there is no need to look at closer distances. In this way considerable time can be saved.

The implemented method allows several other ways of restricting the search, by using a priori information about the scene. A minimum distance and a maximum distance can be specified, and the search will occur only on the portion of the line segments corresponding to this distance range (and within Picture 2, of course). Also, an approximate ground plane can be specified, and the search will be inhibited for disparities that correspond to points that are below this plane by more than a specified height. If desired, a match will not be attempted for any point in Picture 1 which, if on this plane or both above this plane and at an infinite distance, would project outside of Picture 2. All of these restrictions save computation time and tend to prevent incorrect matches, but of course they may also cause correct matches to be missed if the a priori assumptions are not correct.

In areas of low information content, the noise suppression ability of the high-resolution correlator often allows useful results to be obtained. However, if the information content of the picture in certain areas is too low, the correlator indicates this fact by producing very large values for the standard deviations of the two position coordinates. In such a case, it might have been desirable to inhibit the searching to save computer time, but even if this is not done, the results are still as good as the standard deviations indicate. (Actually, the correct test to indicate no useful information is to propagate the match accuracy as indicated by its covariance matrix into the computed stereo disparity for this point, as described in Chapter 5, and to check the size of the resulting standard deviation relative to the magnitude of the disparity. Both standard deviations in the film plane might be large, but if only one eigenvalue of the covariance matrix is large, an accurate disparity, and hence distance, can still be computed for this point unless the corresponding eigenvector is almost parallel to the projected line segment.)

The implemented version of the method contains a way of inhibiting (if desired) the search where it is unlikely that useful information will be obtained. It operates as follows. The standard deviation about the mean of the Picture 1 data within the current match window area for which a match might be searched is computed. Then the F test that is performed in the correlator, as described in Chapter 2 is performed, except that instead of the variance based upon the residuals that is used in the correlator, this standard deviation about the mean is used. This F test gives the probability that the noise level in the data could have given rise to the observed variation in the data to be matched. If this probability is low, then there must be considerable variation above the noise level, and thus the correlator should be able to match this point. If the probability is high, the data may be lost in the noise, and thus a search for a match can be inhibited. A probability threshold of 0.1 perhaps is appropriate.

In deciding whether to accept matches as described below, a tolerance is used in checking the agreement of disparity in adjacent matched areas. Ideally, the tolerance should also take into account the accuracy of the difference in the matches as given by the sum of the two covariance matrices from the correlator (perhaps accepting anything within three standard deviations in addition to a given tolerance). However, this has not been implemented, and currently a constant tolerance is used.

When a window-sized area in Picture 1 has been selected for an attempted match, the first thing to do is to try to avoid a search by seeing if a good match agreeing approximately with neighboring points already matched can be made. To do this, the three adjacent areas in the previous column just to the left (the last column processed) are inspected. (These are the areas directly to the left and diagonally to the left both up and down.) If at least two of these have been successfully matched and if their relative matching positions in Picture 2 all agree within the tolerance described above (or twice this tolerance when comparing the top and bottom of the three areas), then the correlator is applied to the area in question, with the search window in Picture 2 centered on the position corresponding to the average matching position of these two or three neighbors (suitably displaced according to the shift to the right and up or down from the position of the neighboring areas in Picture 1). If the probability computed by the correlator is greater than some threshold (0.1 is used currently), this match is accepted and no search is done. However, if the computed matching point in Picture 2 has already been matched, the current match is accepted only if its probability is greater than that of the old match, in which case the old match is deleted. The tolerance used for checking whether these matched points in Picture 2 coincide is half of the minimum of the match window width and the step size in Picture 1 (which normally are equal).

If the above trial match is not accepted, the search is done in the following manner. The point at the center of the match window in Picture 1 is corrected for distortion as described in Chapter 1 and is projected into a line segment in Picture 2 as described in

Appendix B. Working from the infinity point towards lesser distances, points are chosen along this line and are distorted to represent points in the actual picture instead of in a central projection. Previously successfully matched points within the tolerance described above are skipped, and the correlator is applied with its search window centered on the remaining points. A good spacing to use for these points would be half of the search window width, which would produce sufficient overlap so that the computed matched point will not be forced to be near the edge of the search window. However, in order to speed up the program a main step size equal to the search window width is used. But if the probability computed by the correlator for one of these trials is greater than 0.05 or greater than half of the greatest probability found so far in this search, then another trial is made a half step ahead if the computed position was in the front quarter of the search window, or a half step behind if it was in the back quarter. In this way, if the correct matching position occurs approximately on the boundary between two successive search windows (without overlap), it will be found more nearly centered within an overlapping search window produced in this manner, thus avoiding the loss of accuracy from the truncation at the edge of the search window.

Of all of the matches produced by the correlator in the above search, the one with the highest probability is tentatively selected. This is checked for agreement within the specified disparity tolerance with neighboring matched areas, including all three neighboring areas in the previous column, which may have accepted matches, and the two areas directly above and below in this column, which may have an accepted match or tentative matches produced in this manner. If there is agreement with at least one of these neighbors, and if of the two matches consisting of this neighbor and the current match under consideration both probabilities are at least 0.01 and either probability is at least 0.1, the current match is accepted. Otherwise, this area is left unmatched.

It was originally intended to have the method try further in case no match was accepted above, by comparing the results from those points in the search for which the correlator produced less then maximum probability with those from the adjacent search (above or below), as mentioned in Gennery [1977]. If an adjacent pair could be found which agreed closely in disparity and both of which had reasonably high probability, this pair of matches could be accepted. This additional feature was tried, but using it considerably increased the number of incorrect matches. Therefore, it was not adopted. Also, in order to try to reduce the number of incorrect matches, a feature was tried which accepted the best match in a search only if its probability was at least twice as great as the second best. But this resulted in the loss of a great number of correct matches in low-contrast areas, so it too was rejected.

One more refinement in the above method remains to be discussed. Because of perspective distortion, a match window in one picture will not match exactly the corresponding match window in the other picture. In many outdoor scenes a large portion of the points are on the ground, and these points may be very important in finding the ground surface, as described in Chapter 6. Also, because the ground makes a large angle with the camera focal plane in a roughly horizontal camera, areas on the ground often suffer a large amount of perspective distortion. Furthermore, not only does this distortion affect the correlator, but it also may cause adjacent matches (above and below each other) to disagree by more than the disparity tolerance, unless this tolerance is made very generous. Therefore, provision has been made for using the correlator both straight and with a predistorted match window corresponding to the distortion that would occur if the point is on the a priori approximate ground plane, and for compensating the check of agreement of neighbors correspondingly for this distortion. Of course, the correlator could include a general search over distortion, but this would be very time-consuming. Including a special distortion correction for the ground at a cost of about doubling the computation time is justified in some cases by the importance of the ground and the large distortion that it may undergo. (Clark Thompson [1975] also has suggested such a distortion correction. Mori *et al.* [1973] use a more elaborate prediction-correction technique to handle general perspective distortion.)

If the ground is a plane, the perspective distortion of the match window between pictures is a constant skew distortion, for all points on the ground. Only a special case has been implemented, in which it is assumed that all angles of the camera model except azimuth are zero (that is, the two cameras have the same orientation, and the stereo axis projects into the film plane as a line parallel to the x axis) and that the ground plane is parallel to the x axis. In this case the amount of this skew is given by $(r \sin \alpha_1 \cos \lambda)/h$, where α_1 is the azimuth from camera 1 to camera 2 relative to the camera axis, λ is the tilt of the ground relative to the axis, r is the distance between the cameras, and h is the height of camera 1 above the ground plane. The skew given by this formula is the tangent of the angle that a straight line on the ground would make with the y axis when projected into the camera 1 film plane, if it is parallel to the y axis when projected into the camera 2 film plane.

Thus the algorithm as described above has been modified so that the following computations are included, when desired. When the correlator is applied to a portion of the picture which could be on the a priori ground plane (that is, the point is not above the a priori horizon in Picture 1), it is applied both with a normal match window and with a match window which has been distorted into a parallelogram in picture 1 to correspond to the square match window in picture 2, according to the skew as computed above. (The values used within this skewed window are obtained by linear interpolation from the original picture values, although if the interpolating version of the correlator were used, it could do this interpolation itself.) In checking for agreement with neighboring areas, the x coordinates of the points are shifted according to the skew when results from the skewed window are used, but are used unchanged when the normal window is used. In the preliminary match to avoid a search (trying a match with approximately the same disparity as neighbors already matched), the result with the greater probability of the two is used. In the search along the projected line segment, the two results for each trial position simply double the effective number of trials, and the best result is selected as before (based on the probability and agreement with neighbors).

Figures 4-1 and 4-2 show the results of applying the matching algorithm to the Mars pictures described in Appendix C. Match windows and search windows were both 8 pixels by 8 pixels. Each dot superimposed on the left picture (Figure 4-1) is at the center of an 8-by-8 area that was successfully matched. An error ellipse is shown centered on the point in the right picture (Figure 4-2) to which each of these points was matched. The ellipses shown represent the three-standard deviation limits. If a normal distribution of position errors is assumed (actually not a good assumption for this kind of error), about one out of ninety points would be expected to have the true matching position outside of the ellipse. Lines connect points in Figure 4-2 which match points forming a vertical column in Figure 4-1. Dashed lines bridge gaps caused by unmatched areas in the left picture. It can be seen that three obviously incorrect matches were made (in the lower right fourth of the picture), but the rest appear to be more or less reasonable.

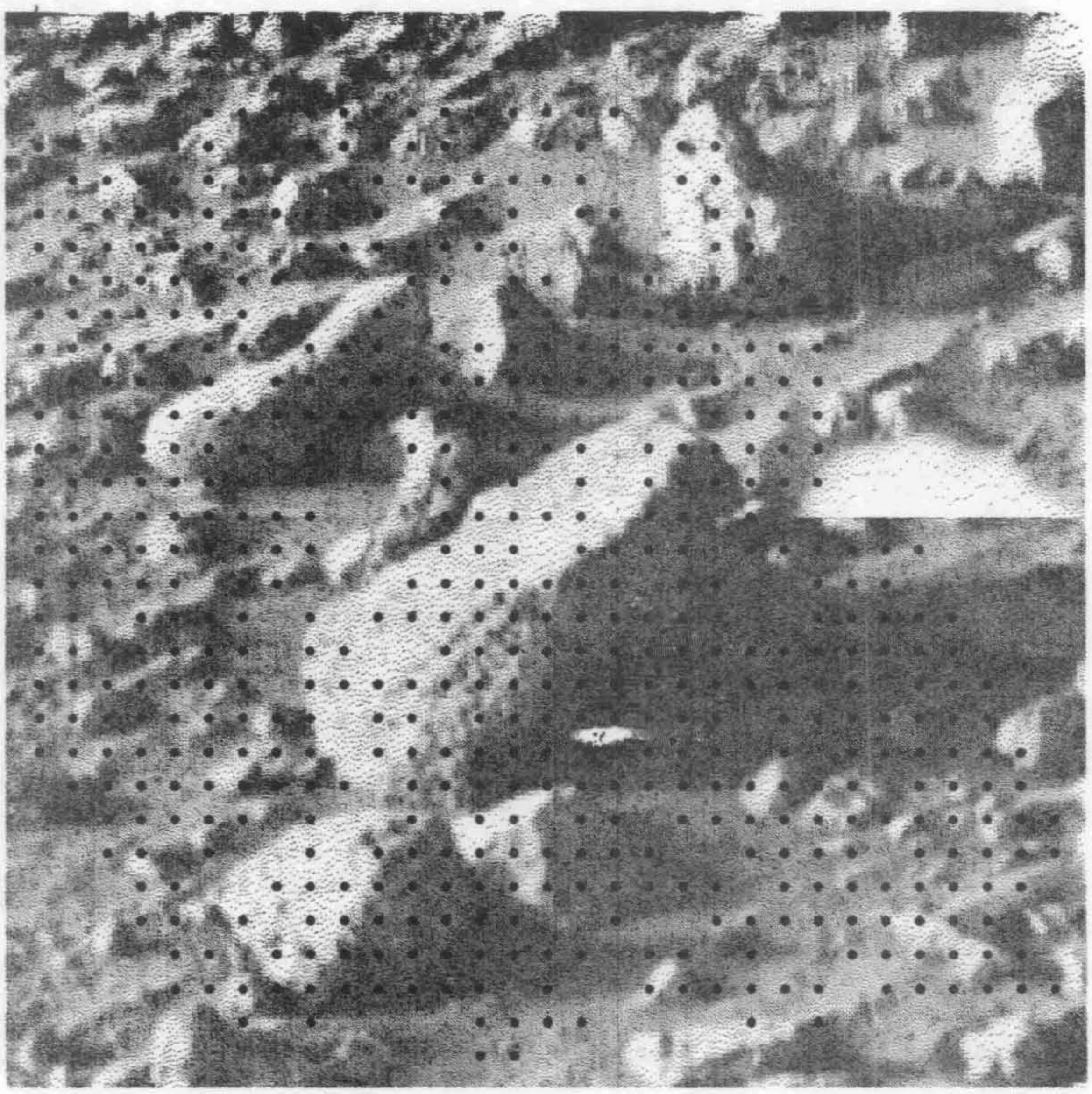

Figure 4-1. Points found in left picture.

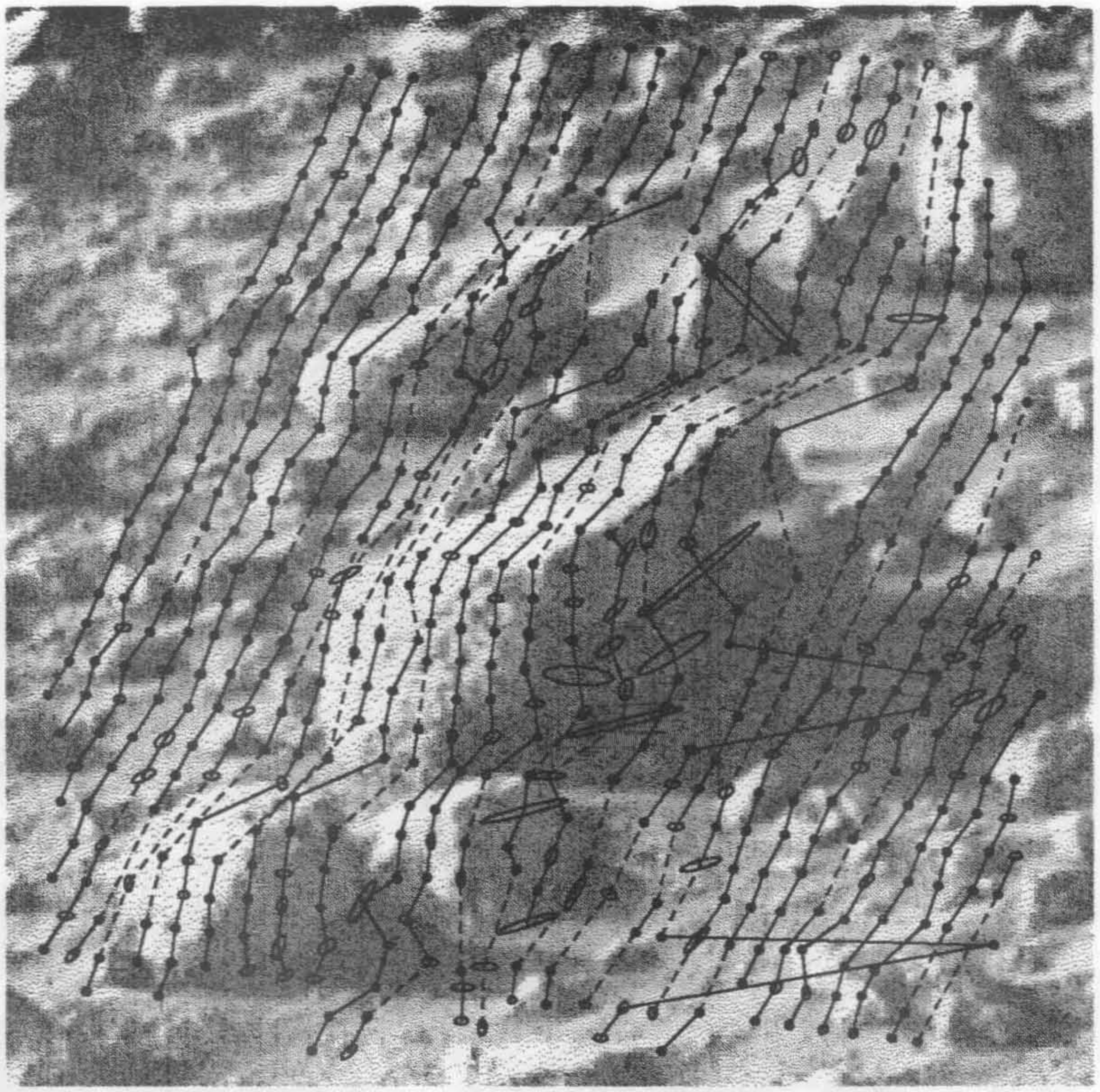

Figure 4-2. Matched points in right picture, showing 3σ error ellipses, with lines connecting points corresponding to columns in Figure 4-1.

Chapter 5
DISTANCE COMPUTATION

Once a pair of matching stereo points has been found, it is usually considered to be fairly trivial to compute the distance to the corresponding point in three-dimensional space, if the stereo camera model is known, as in Hannah [1974]. However, here this process is complicated by two facts. First, the available information about the accuracy of each point (obtained from the correlator) implies that the optimum matching point in the film plane in general is not the point on the projection line (of the Picture 1 point into Picture 2) that is nearest to the matching point found by the correlator. Computing the stereo disparity corresponding to the optimum matching point requires the use of the two-by-two covariance matrix produced by the correlator. (The nearest point is optimum only if this matrix is a scalar matrix or one of its eigenvectors is perpendicular to the line.) Second, it is desired not only to compute the distance but also to compute its accuracy, by propagating the accuracy estimates of the matching point and of the camera model into the distance. As is usually the case, this error propagation computation involves considerably more effort than the distance computation itself. It is complicated further here by the fact that more than one type of accuracy estimate may be desired, depending on to what extent the effects of camera model error are to be included.

5.1 Matching Point

The computations described in the previous chapters have produced, for some point x_1, y_1 (corrected for distortion) in the Camera 1 image plane, an estimate x_2 and y_2 (corrected for distortion) of the matching position in the Camera 2 image plane and its accuracy represented by σ_x^2, σ_y^2, and σ_{xy}. The accuracy estimates may contain both a random component, obtained from the correlator or other information independently for each point, and a systematic component, which might be obtained from the additional variance adjustment in the camera model solution or from other information. From the camera model information, the projection of the Camera 1 point into the Camera 2 film plane can be computed as described in Appendix B to produce a line segment represented by the infinity point x_0, y_0 and the direction cosines c_x and c_y. It is now desired to use this information to compute the optimum matching position x_p, y_p in the Camera 2 image plane, considering both the estimate x_2, y_2 from the correlator and the projection of x_1, y_1 according to the camera model information.

First consider the camera model to be known exactly, so that there is no uncertainty in x_0, y_0, c_x, and c_y. The relationships are shown in Figure 5-1. The ellipses are contours of equal probability density from the distribution given by σ_x^2, σ_y^2 and σ_{xy} (including both random and systematic error), assuming a normal distribution. The quantity v is the same

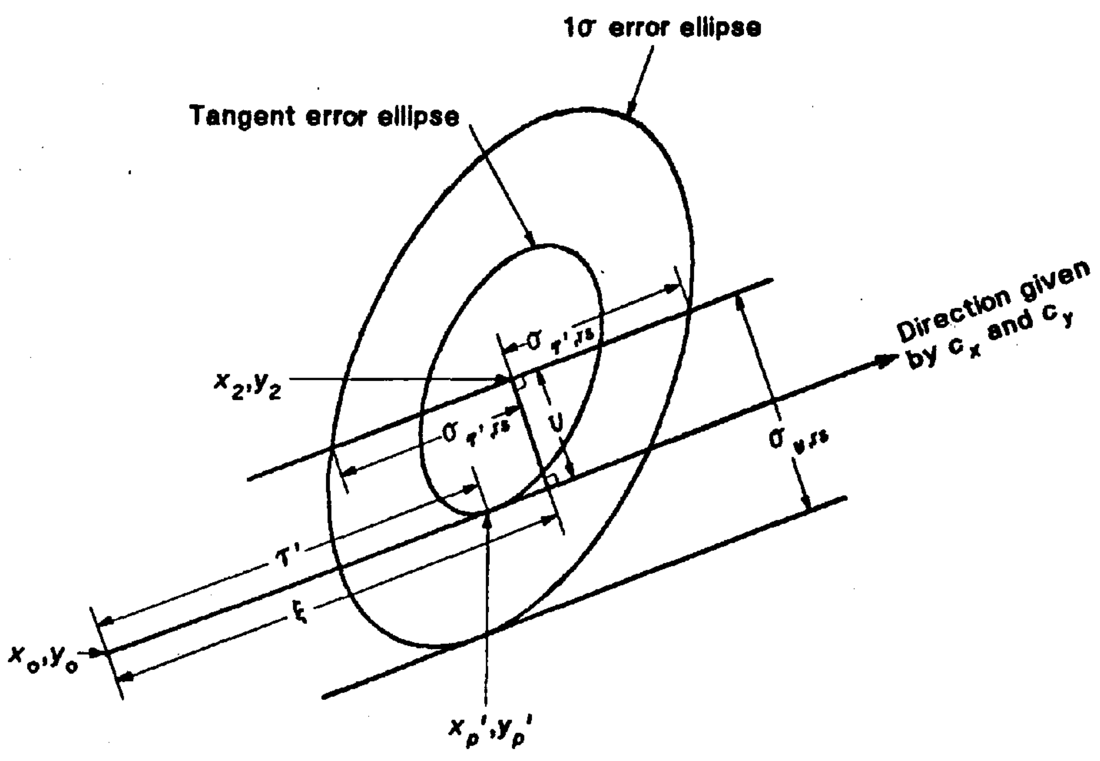

Figure 5–1. Determination of optimum matching point, neglecting camera model error.

as the discrepancy used in the camera model adjustment. The quantity ξ is the stereo disparity (defined here as being relative to the infinity point) usually used, as in Hannah [1974]. However, τ' is the quantity used here for the disparity. It is the distance in the Camera 2 image plane from the infinity point to the tangent point of one of the ellipses and the projection line of x_1, y_1. (The prime is used because there may be a correction to be applied to this quantity when the effects of the uncertain camera model are considered.)

Since c_x and c_y are the cosines of the angles of the projection line with the x and y axes, the following relationships hold:

$$\begin{aligned} \xi &= c_x(x_2-x_0) + c_y(y_2-y_0) \\ \upsilon &= c_x(y_2-y_0) - c_y(x_2-x_0) \end{aligned} \qquad (5.1\text{-}1)$$

Also,

$$\begin{aligned} x_p &= x_0 + c_x\tau \\ y_p &= y_0 + c_y\tau \end{aligned} \qquad (5.1\text{-}2)$$

where the prime has been dropped, since (5.1-2) holds for both corrected and uncorrected values.

Now the value of τ' must be determined. One way to do this is to consider x_2 and y_2 to be observations on the quantities x_p and y_p, which are functions of the parameter τ according to (5.1-2). A generalized least-squares solution can be done for τ, with weights derived from the values σ_x^2, σ_y^2, and σ_{xy}. Since (5.1-2) is linear in τ, equation (A.1-17) can be used exactly, and no iteration is required. As described in Appendix A, the proper weight matrix to use is the inverse of the covariance matrix of observations. Thus the unique elements of the weight matrix are

$$\begin{aligned} w_{xx} &= \frac{\sigma_y^2}{\sigma_x^2\sigma_y^2 - \sigma_{xy}^2} \\ w_{xy} &= \frac{-\sigma_{xy}}{\sigma_x^2\sigma_y^2 - \sigma_{xy}^2} \\ w_{yy} &= \frac{\sigma_x^2}{\sigma_x^2\sigma_y^2 - \sigma_{xy}^2} \end{aligned} \qquad (5.1\text{-}3)$$

where the variances and covariance include both random and systematic error. Upon making the appropriate substitutions derived from (5.1-2), equation (A.1-20) becomes

$$\sigma^2_{\tau',rs} = \left(\begin{bmatrix} c_x & c_y \end{bmatrix} \begin{bmatrix} w_{xx} & w_{xy} \\ w_{xy} & w_{yy} \end{bmatrix} \begin{bmatrix} c_x \\ c_y \end{bmatrix}\right)^{-1}$$

$$= \frac{1}{c_x^2 w_{xx} + 2c_x c_y w_{xy} + c_y^2 w_{yy}} \qquad (5.1\text{-}4)$$

and equation (A.1-17) becomes

$$\tau' = \sigma^2_{\tau',rs} \begin{bmatrix} c_x & c_y \end{bmatrix} \begin{bmatrix} w_{xx} & w_{xy} \\ w_{xy} & w_{yy} \end{bmatrix} \begin{bmatrix} x_2 - x_0 \\ y_2 - y_0 \end{bmatrix}$$

$$= \frac{c_x w_{xx}(x_2-x_0) + c_x w_{xy}(y_2-y_0) + c_y w_{xy}(x_2-x_0) + c_y w_{yy}(y_2-y_0)}{c_x^2 w_{xx} + 2c_x c_y w_{xy} + c_y^2 w_{yy}} \qquad (5.1\text{-}5)$$

(The subscript "rs" indicates that this estimate of accuracy of τ' includes random and systematic error from the point x_2, y_2 but does not include camera model error.)

This equation for τ' needs to be in a form more convenient for error propagation. By using the fact that $c_x^2 + c_y^2 = 1$, some algebraic manipulation can transform (5.1-5) into

$$\tau' = c_x(x_2-x_0) + c_y(y_2-y_0) + \frac{c_x c_y(w_{yy}-w_{xx}) + (c_x^2-c_y^2)w_{xy}}{c_x^2 w_{xx} + 2c_x c_y w_{xy} + c_y^2 w_{yy}}[c_x(y_2-y_0) - c_y(x_2-x_0)] \qquad (5.1\text{-}6)$$

Now let

$$\rho = \frac{c_x c_y(w_{yy}-w_{xx}) + (c_x^2-c_y^2)w_{xy}}{c_x^2 w_{xx} + 2c_x c_y w_{xy} + c_y^2 w_{yy}} \qquad (5.1\text{-}7)$$

Then, by using (5.1-1) and (5.1-7), equation (5.1-6) can be rewritten as

$$\tau' = \xi + \rho v \qquad (5.1\text{-}8)$$

These results can be expressed in terms of the variances and covariance instead of the weights. Substituting (5.1-3) into (5.1-4) and (5.1-7) produces

$$\sigma^2_{\tau',rs} = \frac{\sigma_x^2 \sigma_y^2 - \sigma_{xy}^2}{c_x^2 \sigma_y^2 - 2c_x c_y \sigma_{xy} + c_y^2 \sigma_x^2} \qquad (5.1\text{-}9)$$

and

$$\rho = \frac{c_x c_y(\sigma_x^2-\sigma_y^2) + (c_y^2-c_x^2)\sigma_{xy}}{c_x^2\sigma_y^2 - 2c_x c_y \sigma_{xy} + c_y^2\sigma_x^2} \tag{5.1-10}$$

Thus using (5.1-1), (5.1-10), (5.1-8), and (5.1-2) produces the desired matching point.

The accuracy estimate of τ' from (5.1-9) can be propagated through (5.1-2) as shown in the next section to produce the effect of combined random and systematic point error on the results. However, if only the effect of random error is desired, the value of $\sigma^2_{\tau',r}$ must be computed to be used instead. In order to do this, first (5.1-6) is rewritten as follows by substituting (5.1-7) and rearranging:

$$\tau' = (c_x - \rho c_y)(x_2 - x_o) + (c_y + \rho c_x)(y_2 - y_o) \tag{5.1-11}$$

Then, since ρ is independent of x_2 and y_2, the error propagation from x_2 and y_2 to τ' produces

$$\sigma^2_{\tau',r} = (c_x - \rho c_y)^2\sigma^2_{x,r} + 2(c_x - \rho c_y)(c_y + \rho c_x)\sigma_{xy,r} + (c_y + \rho c_x)^2\sigma^2_{y,r} \tag{5.1-12}$$

where the subscript "r" denotes that only random error is included. (If both random and systematic error were included here, substituting (5.1-10) into (5.1-12) and simplifying would produce (5.1-9). However, this simplification does not occur for the random error, because ρ is always computed from the total point error according to (5.1-10).)

Now consider the effect of uncertainty in the camera model, represented by S_C, the covariance matrix of the camera model parameters defined in Appendix B. There are two cases to consider, according to whether the point in question (x_2, y_2) was used in a camera model adjustment which produced the camera model being used.

If this point was used in determining the camera model, then the values x_o, y_o, c_x, and c_y used above take into account the information in this point, as represented by x_2, y_2, σ_x^2, σ_y^2, and σ_{xy}. Thus the matching point x_p, y_p computed above represents the best compromise between the information in this point and the other information which went into the camera model determination. Therefore,

$$\begin{aligned} \tau &= \tau' \\ \sigma^2_{\tau,r} &= \sigma^2_{\tau',r} \\ \sigma^2_{\tau,rs} &= \sigma^2_{\tau',rs} \end{aligned} \tag{5.1-13}$$

However, the fact that there is uncertainty in the camera model solution causes components of uncertainty in the disparity in addition to that represented by $\sigma^2_{\tau,rs}$.

Since both systematic point error and camera model error will affect nearby points in related ways, causing correlated errors in these points, the component of error which affects points independently is due to $\sigma^2_{\tau,r}$. Thus propagating this error from τ into the final results will produce what is called here "independent error."

To compute the total error, the partial derivatives of the the various above quantities relative to the camera model parameters must be computed, so that by a linear approximation the accuracy estimates can be propagated. However, because of the different shapes and orientations that the error ellipses for different points can have, certain changes in the camera model parameters can cause changes in nearby points that are different but correlated. These effects occur by the effects of the camera model parameters on the quantities ρ and υ. (The effects of the camera model parameters on ξ and on x_p and y_p through their functional dependence of x_0, y_0, c_x, and c_y explicitly indicated in (5.1-2) produce practically the same effect on nearby points.) Combining this type of error (from ρ and υ) with the random point error produces what is called here "relative error," denoted by the subscript "rel".

By differentiating (5.1-1), (5.1-2), (5.1-8), and (5.1-10) the desired partial derivatives can be obtained. These can be expressed as follows in terms of the partial derivatives of x_0, y_0, c_x, and c_y obtained as described in Appendix B:

$$\frac{\partial \xi}{\partial g} = (x_2-x_0)\frac{\partial c_x}{\partial g} + (y_2-y_0)\frac{\partial c_y}{\partial g} - c_x\frac{\partial x_0}{\partial g} - c_y\frac{\partial y_0}{\partial g}$$

$$\frac{\partial \upsilon}{\partial g} = (y_2-y_0)\frac{\partial c_x}{\partial g} - (x_2-x_0)\frac{\partial c_y}{\partial g} - c_x\frac{\partial y_0}{\partial g} + c_y\frac{\partial x_0}{\partial g}$$

$$\frac{\partial \rho}{\partial g} = \frac{[c_y(\sigma_x^2-\sigma_y^2) - 2c_x\sigma_{xy} + 2\rho(c_y\sigma_{xy}-c_x\sigma_y^2)]\frac{\partial c_x}{\partial g} + [c_x(\sigma_x^2-\sigma_y^2) + 2c_y\sigma_{xy} + 2\rho(c_x\sigma_{xy}-c_y\sigma_x^2)]\frac{\partial c_y}{\partial g}}{c_x^2\sigma_y^2 - 2c_xc_y\sigma_{xy} + c_y^2\sigma_x^2} \tag{5.1-14}$$

$$\frac{\partial \tau}{\partial g} = \frac{\partial \xi}{\partial g} + \rho\frac{\partial \upsilon}{\partial g} + \upsilon\frac{\partial \rho}{\partial g}$$

$$\left(\frac{\partial \tau}{\partial g}\right)_{\text{rel}} = \rho\frac{\partial \upsilon}{\partial g} + \upsilon\frac{\partial \rho}{\partial g}$$

$$\frac{\partial x_p}{\partial g} = \frac{\partial x_0}{\partial g} + c_x\frac{\partial \tau}{\partial g} + \tau\frac{\partial c_x}{\partial g}$$

$$\frac{\partial y_p}{\partial g} = \frac{\partial y_0}{\partial g} + c_y\frac{\partial \tau}{\partial g} + \tau\frac{\partial c_y}{\partial g}$$

where g denotes any one of the camera model parameters defined in Appendix B. The partial derivatives from the last three equations of (5.1-14) will be used in the error propagation in the next section.

If this point was not used in determining the camera model, there is no unequivocally best thing to do. (The only optimum thing to have done would have been to have included all of the points in the camera model solution. However, this may have been impractical because of the time required. For example, after the dense matching of points have been computed as described in Chapter 4, all of these points could be used in a new camera model solution. But if the camera model is already known sufficiently accurately, the additional computing may not be worthwhile.) One thing to do would be simply to use the same solution described above. This has the advantage that the solution for each point is based on the same camera model, and thus the independent error is mimimum. However, it may be desired to produce the best compromise between the information in this point and the information in the camera model (as far as this point is concerned, without considering any other points). This approach reduces the total error but increases the independent error, compared to the previous method. Depending on the circumstances, it may either increase or decrease the relative error. (We are speaking here of the variances, that is the expected squares of the errors, not the actual values of the errors for a given point.) A compromise is possible which reduces both the relative error and total error compared to the first method (although it does not reduce the total error as much as the second method) and is simpler than the second method. This third method includes the effects of changing the camera model only insofar as it affects the point x_p, y_p but does not consider the effects of moving the infinity point x_0, y_0 closer to or further from point x_p, y_p (along the back-projection line). (Because of correlation between the camera model errors parallel and perpendicular to this line, shifting the line sideways to improve agreement with the point x_2, y_2 would cause such movement parallel to the line. Since this movement would be different for every point, it would make the relative error worse.) Since relative error is usually the most important error, this third method is used in the implemented program, and it will now be described.

Consider the quantity v, which is the perpendicular distance from the point x_2, y_2 to the back-projection line. Two variances of this quantity will be considered. First, $\sigma^2_{v,rs}$ includes only the effect of error in the point x_2, y_2 (both random and systematic). It is shown in Figure 5-1, and its equation can be easily derived from the equation for v in (5.1-1) to be

$$\sigma^2_{v,rs} = c_x^2 \sigma_y^2 - 2 c_x c_y \sigma_{xy} + c_y^2 \sigma_x^2 \qquad (5.1\text{-}15)$$

Second, $\sigma^2_{v,c}$ includes only the effect of camera model error. It can be computed from the covariance matrix of camera model error S by using the partial derivatives of v with respect to the camera model computed according to (5.1-14). If these partial derivatives are assembled into the row matrix $\frac{\partial v}{\partial G}$, then

$$\sigma^2_{v,c} = \frac{\partial v}{\partial G} \, S \left(\frac{\partial v}{\partial G} \right)^T \qquad (5.1\text{-}16)$$

Note that υ is the amount by which this point disagrees with the camera model. Thus a compromise position of agreement between these two pieces of information can be obtained by performing a weighted average of them, with the weights inversely proportional to these two variances. (This is correct under a linear approximation only, since the actual variance due to the camera model varies with the position in the image plane.) Therefore, the compromise point is moved from point x_2, y_2 by a fraction k of the distance towards the back-projection line computed from the camera model, and thus it is at a perpendicular distance of $(1-k)\upsilon$ from this line, where

$$k = \frac{\sigma^2_{\upsilon,rs}}{\sigma^2_{\upsilon,rs} + \sigma^2_{\upsilon,c}} \tag{5.1-17}$$

Shifting the point by the distance $k\upsilon$ in the direction perpendicular to the line causes it to shift by the distance $\rho k\upsilon$ parallel to the line, because of the correlation in the errors in the point, according to (5.1-8). Also, the back projection line moves a distance $(1-k)\upsilon$ towards the point, so that the compromise point lies on the compromise line. However, as stated above, in this method we do not want to change the camera model. Instead, the compromise point will be projected perpendicularly onto the line computed from the given camera model. Doing this does not affect the stereo disparity or the distance, under a linear approximation.

Therefore, the stereo disparity used is

$$\begin{aligned} \tau &= \xi + k\rho\upsilon \\ &= \tau' - (1-k)\rho\upsilon \end{aligned} \tag{5.1-18}$$

instead of (5.1-13). The partial derivatives of the disparity with respect to the camera model parameters are

$$\begin{aligned} \frac{\partial \tau}{\partial g} &= \frac{\partial \xi}{\partial g} + k\rho\frac{\partial \upsilon}{\partial g} + k\upsilon\frac{\partial \rho}{\partial g} \\ \left(\frac{\partial \tau}{\partial g}\right)_{\text{rel}} &= k\rho\frac{\partial \upsilon}{\partial g} + k\upsilon\frac{\partial \rho}{\partial g} \end{aligned} \tag{5.1-19}$$

instead of the corresponding equations in (5.1-14). (The partial derivatives of k do not have to be included. Their effects are negligible compared to the other effects.)

The equation for the variance of τ due to point error can be derived from the second form of (5.1-18) in a straightforward way in terms of the variance of τ' (obtained from (5.1-9) or (5.1-12)), the variance of υ (obtained from (5.1-15) or its equivalent for random error), and the covariance of τ' and υ. (Note that ρ is not a function of the point x_2, y_2. The effect of the point on k is neglected as being negligible.) The equation for the random covariance can be derived from (5.1-12) and (5.1-1) to be

$$\sigma_{\tau'\upsilon,r} = -(c_x - \rho c_y)c_y\sigma^2_{x,r} + (c_x - \rho c_y)c_x\sigma_{xy,r} - (c_y + \rho c_x)c_y\sigma_{xy,r} + (c_y + \rho c_x)c_x\sigma^2_{y,r} \quad (5.1\text{-}20)$$

A corresponding equation for the covariance from total point error exists, but substituting the expression for ρ from (5.1-10) into it causes it to reduce to zero. Therefore, the variance of disparity from point error in this case (point not used in computing the camera model) is

$$\begin{aligned} \sigma^2_{\tau,r} &= \sigma^2_{\tau',r} + (1-k)^2\rho^2\sigma^2_{\upsilon,r} - 2(1-k)\rho\sigma_{\tau'\upsilon,r} \\ \sigma^2_{\tau,rs} &= \sigma^2_{\tau',rs} + (1-k)^2\rho^2\sigma^2_{\upsilon,rs} \end{aligned} \quad (5.1\text{-}21)$$

5.2 Distance

We now have a point x_1, y_1 in Picture 1, point x_p, y_p in Picture 2, and the camera model which relates the two pictures. It is desired to compute the point in three-dimensional space corresponding to these points. This will be the point at which the projections of the two picture points intersect in three-dimensional space. (The projections are guaranteed to intersect, because x_p, y_p was forced to be on the back projection of x_1, y_1 into Picture 2.)

Let $\mathbf{u}$ be the vector from the Camera 1 origin (center of projection) to point x_1, y_1 in the image plane (assumed to be in front of the lens, as explained in Appendix B), let $\mathbf{v}$ be the vector from the Camera 2 origin to point x_p, y_p in the Camera 2 image plane in the same manner, let $\mathbf{r}$ be the vector from the Camera 1 origin to the Camera 2 origin, and let $\mathbf{s}$ be the vector from the Camera 1 origin to the desired point in three-dimensional space. All of these vectors are coplanar, because of the fact that the two projections intersect, as stated above. It is desired to compute $\mathbf{s}$; all other of these vectors are known. Furthermore, let θ be the angle between $\mathbf{r}$ and $\mathbf{v}$, let ϕ be the angle between $\mathbf{u}$ and $\mathbf{v}$, and let f_1 and f_2 denote the principal distances defined in Appendix B. Figure 5-2, which for simplicity assumes that the axes of the cameras lie in the plane of these vectors, illustrates these quantities.

From the law of sines for plane triangles,

$$s = \frac{r \sin \theta}{\sin \phi} \quad (5.2\text{-}1)$$

But the sine of the angle between two vectors is the magnitude of the cross product of the

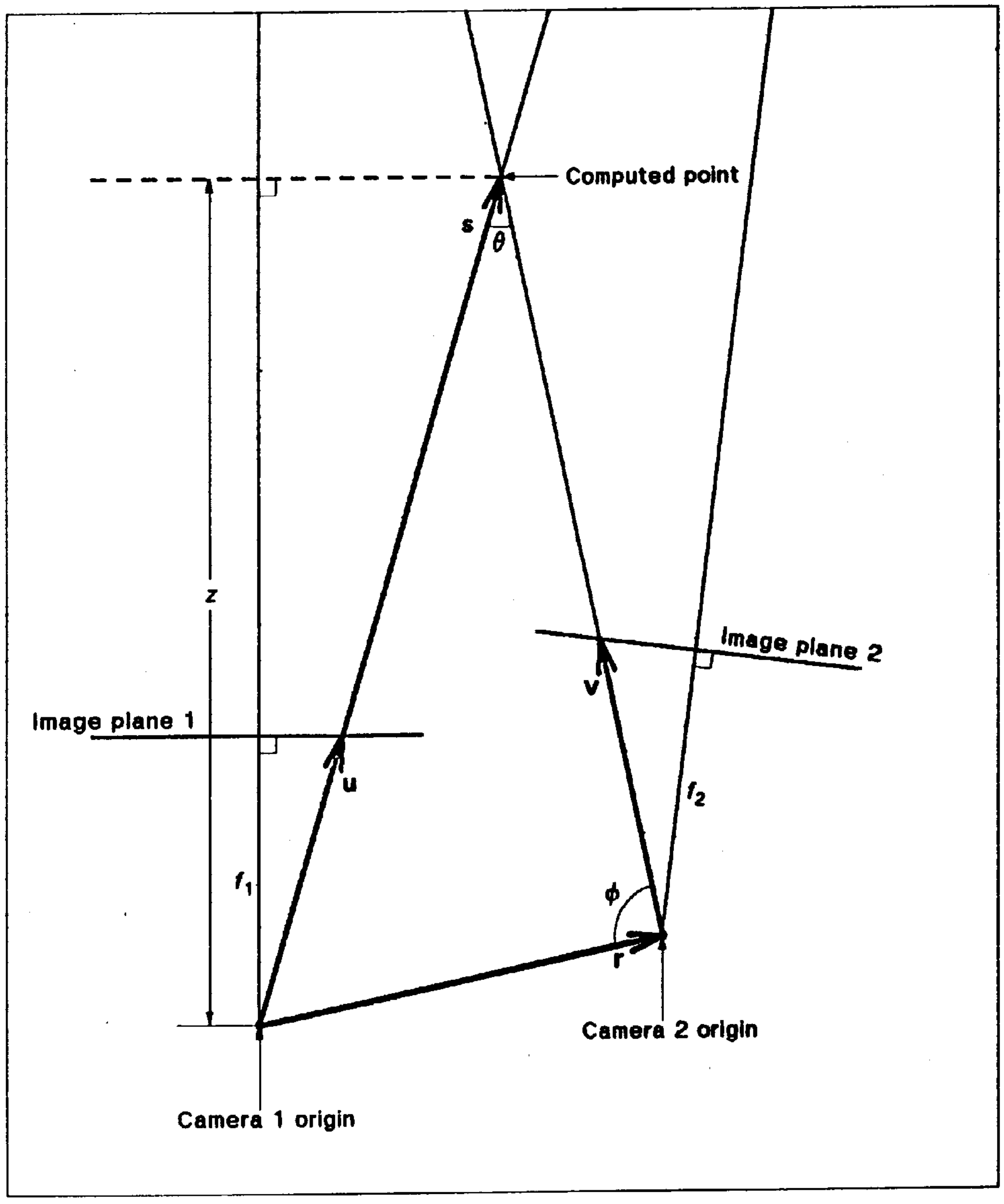

Figure 5-2. Triangulation to compute distance (two-dimensional version).

unit vectors in the same directions. Thus (5.2-1) is equivalent to

$$s = r\,\frac{|\mathbf{r}\times\mathbf{v}|}{rv}\cdot\frac{uv}{|\mathbf{u}\times\mathbf{v}|} = \frac{|\mathbf{r}\times\mathbf{v}|}{|\mathbf{u}\times\mathbf{v}|}\,u \qquad (5.2\text{-}2)$$

Since $\mathbf{u}$ and $\mathbf{s}$ are colinear,

$$\mathbf{s} = \frac{|\mathbf{r}\times\mathbf{v}|}{|\mathbf{u}\times\mathbf{v}|}\,\mathbf{u} \qquad (5.2\text{-}3)$$

Because $\mathbf{r}$, $\mathbf{u}$, and $\mathbf{v}$ are coplanar, the two cross products in (5.2-3) produce parallel vectors. Thus the absolute value operation can be dropped in this equation, and it can be expressed in terms of the ratio of two vectors, as follows:

$$\mathbf{s} = \frac{\mathbf{r}\times\mathbf{v}}{\mathbf{u}\times\mathbf{v}}\,\mathbf{u} \qquad (5.2\text{-}4)$$

Even though the ratio of two vectors is usually not defined, in the case of parallel vectors it is taken to mean the ratio of corresponding components of the two vectors (all of which have the same ratio).

All that we need to compute here is the component of $\mathbf{s}$ parallel to the principal axis of Camera 1, which we denote z. The other two components in the Camera 1 coordinate system can then be easily computed as $x_1 z/f_1$ and $y_1 z/f_1$. Thus, taking this component of both sides of (5.2-3) produces

$$z = \frac{\mathbf{r}\times\mathbf{v}}{\mathbf{u}\times\mathbf{v}}\,f_1 \qquad (5.2\text{-}5)$$

which will be called the "distance" here, rather than using this term for the slant range s.

The vectors needed in (5.2-5) can be expressed in any particular coordinate system for computational purposes; the Camera 2 coordinate system is chosen here. The components of these vectors can be computed by using the unit vector $\mathbf{1}_r$ and the rotation matrix B defined in Appendix B. Thus, in the Camera 2 coordinate system,

$$\mathbf{r} = rB\mathbf{1}_r \qquad \mathbf{u} = B\begin{bmatrix} x_1 \\ y_1 \\ f_1 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} x_p \\ y_p \\ f_2 \end{bmatrix} \qquad (5.2\text{-}6)$$

The partial derivatives of these vectors relative to the camera model parameters are then

$$\frac{\partial \mathbf{r}}{\partial g} = rB\frac{\partial \mathbf{1}_r}{\partial g} + r\frac{\partial B}{\partial g}\mathbf{1}_r \qquad \frac{\partial \mathbf{u}}{\partial g} = \frac{\partial B}{\partial g}\begin{bmatrix} x_1 \\ y_1 \\ f_1 \end{bmatrix} \qquad \frac{\partial \mathbf{v}}{\partial g} = \begin{bmatrix} \frac{\partial x_p}{\partial g} \\ \frac{\partial y_p}{\partial g} \\ 0 \end{bmatrix} \tag{5.2-7}$$

assuming that f_1 and f_2 are not included in the camera parameters. (If uncertainty in the principal distances is to be propagated also, additional partial derivatives relative to f_1 and f_2 are computed in the obvious way.) The partial derivatives of $\mathbf{1}_r$ and B needed above are obtained as described in Appendix B.

In order to compute the distance z by (5.2-5), where a ratio of two parallel vectors is called for we could use the ratio of the absolute values of the vectors, as in (5.2-3). However, in order to keep the computations simple, which is especially important when computing the partial derivatives for the error propagation, the ratio of one of the components is used. The question then is which component to use. In principle, it could be any nonzero component. However, in order to avoid numerical loss of significance, a small component should be avoided. Since the point in the scene always is in front of the image planes of both cameras, the cross products in (5.2-5) never produce vectors perpendicular to the image plane, and the cross product in the numerator is never zero. Therefore, it is guaranteed that either the x or y components must be significantly large, at least in the numerator. Thus what is done in the implemented program is to compute both the x and y components of the numerator, to select whichever is greater in magnitude, and to select the corresponding component of the denominator. Letting p and q be the components of the numerator and denominator actually used, letting the subscripts x, y, and z denote the components of the vectors, noting that the components of $\mathbf{v}$ are x_p, y_p, and f_2, and writing out the cross products in terms of the individual components produce the following relationships. If $|r_y f_2 - r_z y_p| > |r_z x_p - r_x f_2|$,

$$\begin{aligned} p &= r_y f_2 - r_z y_p \\ q &= u_y f_2 - u_z y_p \end{aligned} \tag{5.2-8}$$

Otherwise,

$$\begin{aligned} p &= r_z x_p - r_x f_2. \\ q &= u_z x_p - u_x f_2 \end{aligned} \tag{5.2-9}$$

Then the distance is

$$z = \frac{pf_1}{q} \tag{5.2-10}$$

In order to do the error propagation, the partial derivatives of p and q relative to the camera model parameters, and relative to T with the camera model parameters held constant, need to be computed. These can be obtained by differentiating (5.1-2), (5.2-8), and (5.2-9). If $|r_y f_2 - r_z y_p| > |r_z x_p - r_x f_2|$,

$$\begin{aligned}
\frac{\partial p}{\partial T} &= -r_z c_y \\
\frac{\partial q}{\partial T} &= -u_z c_y \\
\frac{\partial p}{\partial g} &= \frac{\partial r_y}{\partial g} f_2 - \frac{\partial r_z}{\partial g} y_p - r_z \frac{\partial y_p}{\partial g} \\
\frac{\partial q}{\partial g} &= \frac{\partial u_y}{\partial g} f_2 - \frac{\partial u_z}{\partial g} y_p - u_z \frac{\partial y_p}{\partial g}
\end{aligned} \tag{5.2-11}$$

Otherwise,

$$\begin{aligned}
\frac{\partial p}{\partial T} &= r_z c_x \\
\frac{\partial q}{\partial T} &= u_z c_x \\
\frac{\partial p}{\partial g} &= \frac{\partial r_z}{\partial g} x_p + r_z \frac{\partial x_p}{\partial g} - \frac{\partial r_x}{\partial g} f_2 \\
\frac{\partial q}{\partial g} &= \frac{\partial u_z}{\partial g} x_p + u_z \frac{\partial x_p}{\partial g} - \frac{\partial u_x}{\partial g} f_2
\end{aligned} \tag{5.2-12}$$

Also needed are partial derivatives of the distance z, as follows:

$$\begin{aligned}
\frac{\partial z}{\partial T} &= \frac{f_1}{q} \cdot \frac{\partial p}{\partial T} - \frac{pf_1}{q^2} \cdot \frac{\partial q}{\partial T} \\
\frac{\partial z}{\partial g} &= \frac{f_1}{q} \cdot \frac{\partial p}{\partial g} - \frac{pf_1}{q^2} \cdot \frac{\partial q}{\partial g}
\end{aligned} \tag{5.2-13}$$

where the needed partial derivatives of p and q are available from (5.2-11) or (5.2-12). The partial derivatives of z relative to the g's are assembled into the 1-by-5 matrix $\frac{\partial z}{\partial G}$. (It is assumed above that the principal distances are known exactly. Otherwise, additional partial derivatives relative to f_1 and f_2 would be computed and would be used as additional elements in $\frac{\partial z}{\partial G}$, which would be 1-by-7 instead of 1-by-5.)

Then, by using the description of the three different types of error in Section 5.1, and by using the linear approximation rule for the propagation of covariance matrices (premultiplying by the matrix of partial derivatives and postmultiplying by its transpose), the independent, relative, and total variances of the distance z are

$$\begin{aligned} \sigma^2_{z,\mathrm{ind}} &= \left(\frac{\partial z}{\partial \tau}\right)^2 \sigma^2_{\tau,r} \\ \sigma^2_{z,\mathrm{rel}} &= \left(\frac{\partial z}{\partial \tau}\right)^2 \left\{ \sigma^2_{\tau,r} + \left(\frac{\partial \tau}{\partial G}\right)_{\mathrm{rel}} S_G \left(\frac{\partial \tau}{\partial G}\right)^{\mathrm{T}}_{\mathrm{rel}} \right\} \\ \sigma^2_{z,\mathrm{tot}} &= \left(\frac{\partial z}{\partial \tau}\right)^2 \sigma^2_{\tau,rs} + \frac{\partial z}{\partial G} S_G \left(\frac{\partial z}{\partial G}\right)^{\mathrm{T}} + \frac{z^2}{r^2}\sigma^2_r \end{aligned} \qquad (5.2\text{-}14)$$

where the last term for the total variance is for the uncertainty in the inter-camera distance (assumed to be independent of the other camera model parameters), and where S_G is the covariance matrix of the camera model parameters, as previously described.

Note that in (5.2-14) the error propagation from the camera model to the distance is done in a different way for relative error and total error. The error propagation for relative error could have been done in the same way as for total error, by defining partial derivatives $(\partial z/\partial g)_{\mathrm{rel}}$. However, this would have reduced to the form used above. This simplification does not occur for total error, because of the additional effects considered in the partial derivatives of x_p, y_p, p, and q for total error.

The linear approximation for error propagation used in (5.2-14) is very poor when σ_z is nearly as great as or greater than z. This condition indicates that the point actually could be at an infinite distance. (If z from (5.2-10) is negative, the point appears to be beyond infinity.) When the true z is considerably greater than r, it is more accurate to consider σ_z/z^2 (using the values computed as above) to be the standard deviation of the errors in $1/z$ (neglecting the fact that the point cannot actually be beyond infinity).

Chapter 6

GROUND SURFACE FINDER

Once the three-dimensional positions of a large number of points in an outdoor scene have been determined, it is desired to determine which points are on the ground and which are on objects above the ground. This chapter discusses means of computing the ground surface.

6.1 Basic Ground Finder

By taking a sufficiently small portion of the scene the ground can be approximated by a simple surface whose equation can be determined. The procedure which has been implemented assumes in general that the ground surface is a two-dimensional second-degree polynomial (a paraboloid). However, weights can be given to a priori values of the polynomial coefficients, to incorporate any existing knowledge about the ground surface into the solution. For example, the second degree terms can be weighted out of the solution altogether, so that the ground surface reduces to a plane. It usually is wise to use at least a small amount of weight on zero values of the second-degree terms in order to constrain them to reasonably small values. (The next section discusses how the method could be changed to handle large areas.)

To determine a ground surface from a given set of data, a set of criteria which define what is meant by a good ground surface is needed. These include the number of points within tolerance of the surface (the more the better), the number of points which lie beyond tolerance below the surface (the fewer the better, since these would be due to errors such as mismatched points in a stereo pair), and the closeness of the surface coefficients to the a priori values. Note that the number of points above the surface does not matter (other than that it detracts from the number within the surface), because many points can be on objects above the ground. A score for any tentative solution is computed based on these criteria, and the solution with the highest score is assumed to be correct, although a solution with a lower score could be selected by a higher level procedure using more global criteria. The scoring function currently used is

$$v = \frac{n - m}{n + \nu - 2m} - \frac{k^2}{\kappa^2 + k\kappa} - \sum_i \left(\frac{c_i - c_i'}{3\,\sigma_i}\right)^2 \qquad (6.1\text{-}1)$$

where n is the number of points within tolerance of the surface (these points were used to determine the surface by a least-squares fit), ν is the a priori expected number of points in the surface, k is the number of points below the surface by more than the tolerance, κ is the a priori approximate maximum number of points below the surface, the c_i are the coefficients of the fitted surface, c_i' are their a priori values, σ_i are the standard deviations of these a priori values, and m is the number of these coefficients which were adjusted. The numerical value of the score can range from approximately +1 to arbitrarily large negative numbers. Any positive value is considered to represent a satisfactory fit.

Finding the best solution out of all of the possible solutions is a search problem. What is needed is a method which will be likely to find the correct solution without requiring huge amounts of computer time.

One possibility would work as follows. Take all combinations of the points three at a time for the special case of a plane surface (or six at a time for the more general case), fit the surface to each combination, see what points lie within tolerance of the resulting surface, include these in the solution by a least-squares adjustment, and iterate in this manner until a stable set of points is reached for each tentative solution. However, the number of tentative solutions would be approximately proportional to the cube of the number of points for the plane case (or the sixth power for the general case). Therefore, this method would usually be impractical.

The method actually used uses some heuristics to lead the search to the desired solution. It can be divided into two portions.

First, a least-squares solution is done using all of the points. This fit is saved for refinement leading to one tentative solution. Then all points below this fit, but not less than half of the points used in this fit, are selected, and another least-squares fit is done on these points and saved. This process repeats until there are too few points left. (This portion of the algorithm drives downward to find the low surfaces, even though there may a large amount of clutter above them.)

Second, a refinement of each of the above fits is done, rejecting erroneous points and some clutter, in order to find well-defined surfaces. This refinement process is basically an editing process as described in Appendix A. However, to remove points one at a time as is done with the camera model adjustment may be too time consuming because of the large number of points (many of which may need removing). Therefore, at each step, all points lying outside of the criterion are rejected, and all other points are included, for the next fit. However, in order to avoid rejecting too many points at once (which may include the good points), the standard deviation of the points used in the fit about the fitted surface is computed to obtain a threshold for rejecting points. This wholesale selection of points is permissible because the solution does not need to be rechecked after rejecting each point as is done with the camera model, because the adjustment (as described in this section and in most of the alternatives in the next section) is linear. However; the computed standard deviation will change after the points are rejected. Therefore, instead of using a three-standard-deviation limit as is done with the camera model, a one-standard deviation limit is used (but not less than three times the given standard deviation for each point), in order to avoid rejecting too many points at once. (If a few good points are rejected, they will be reinstated on later iterations as long as the process converges to the correct solution.) This process continues until it stabilizes, in which case the score of the result is computed, or until there are too few points in the solution.

The above algorithm can be summarized as follows:

0. Select all of the given points.

1. Save the next fit done according to step 2.

2. Perform a least-squares fit of the surface to the currently selected points.

3. If all current points (and no others) are within tolerance of the fit, save the fit and go to 7.

4. Compute the standard deviation of the current points from the residuals of the fit.

5. Select all points that are within one computed standard deviation or the original tolerance, whichever is greater.

If $n > m$ (that is, the number of current points is greater than the number of coefficients to be adjusted), go to 2.

7. Using the last fit saved according to step 1, select all points that are below that fit, but not less than half of the points used in that fit. (To avoid rejecting more than half, a limit above the fit is increased from zero by an appropriate amount as indicated by a histogram of the residuals.)

8. If $n > m$ and there is a change in the selected points from the last fit saved according to step 1, go to 1.

9. Of those fits saved according to step 3, the one with the greatest score is the preferred solution, with others ranked in order of decreasing score.

In the general case of a paraboloid mentioned above, the height of the ground surface is

$$h = a + bx + cy + dx^2 + exy + fy^2 \tag{6.1-2}$$

where x and y are the horizontal coordinates, and a, b, c, d, e, and f are the coefficients defining the surface, which are to be determined. Since this equation is linear in these coefficients, the iterations required for the nonlinear solution in Appendix A are not

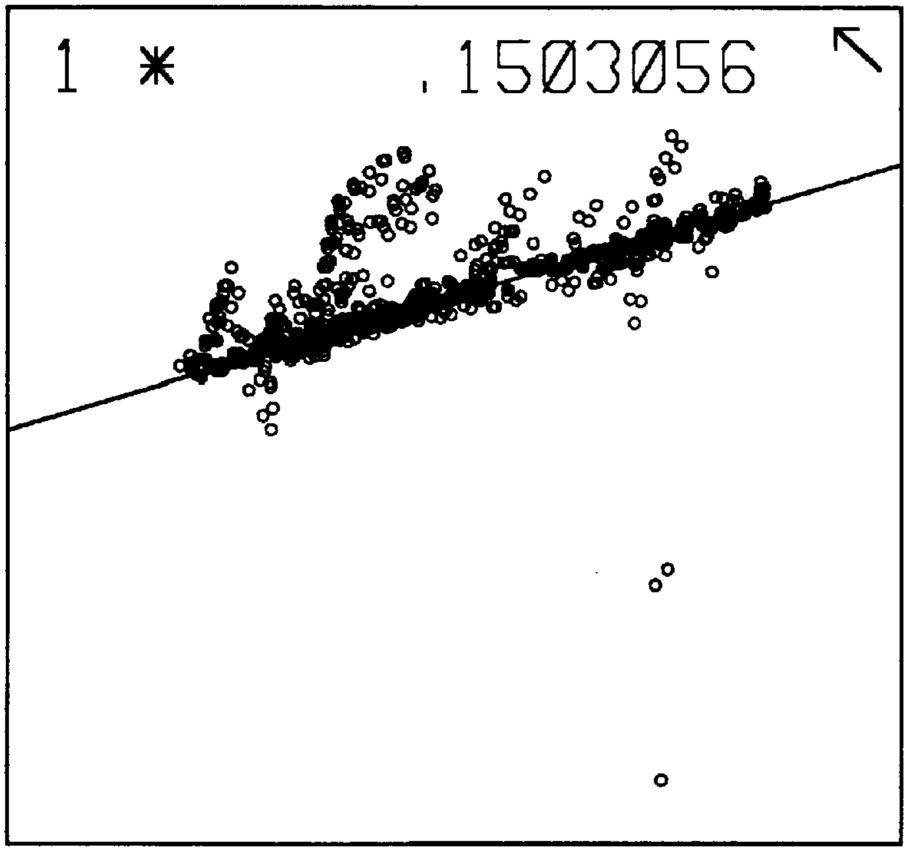

Figure 6-1. Side view of ground fit.

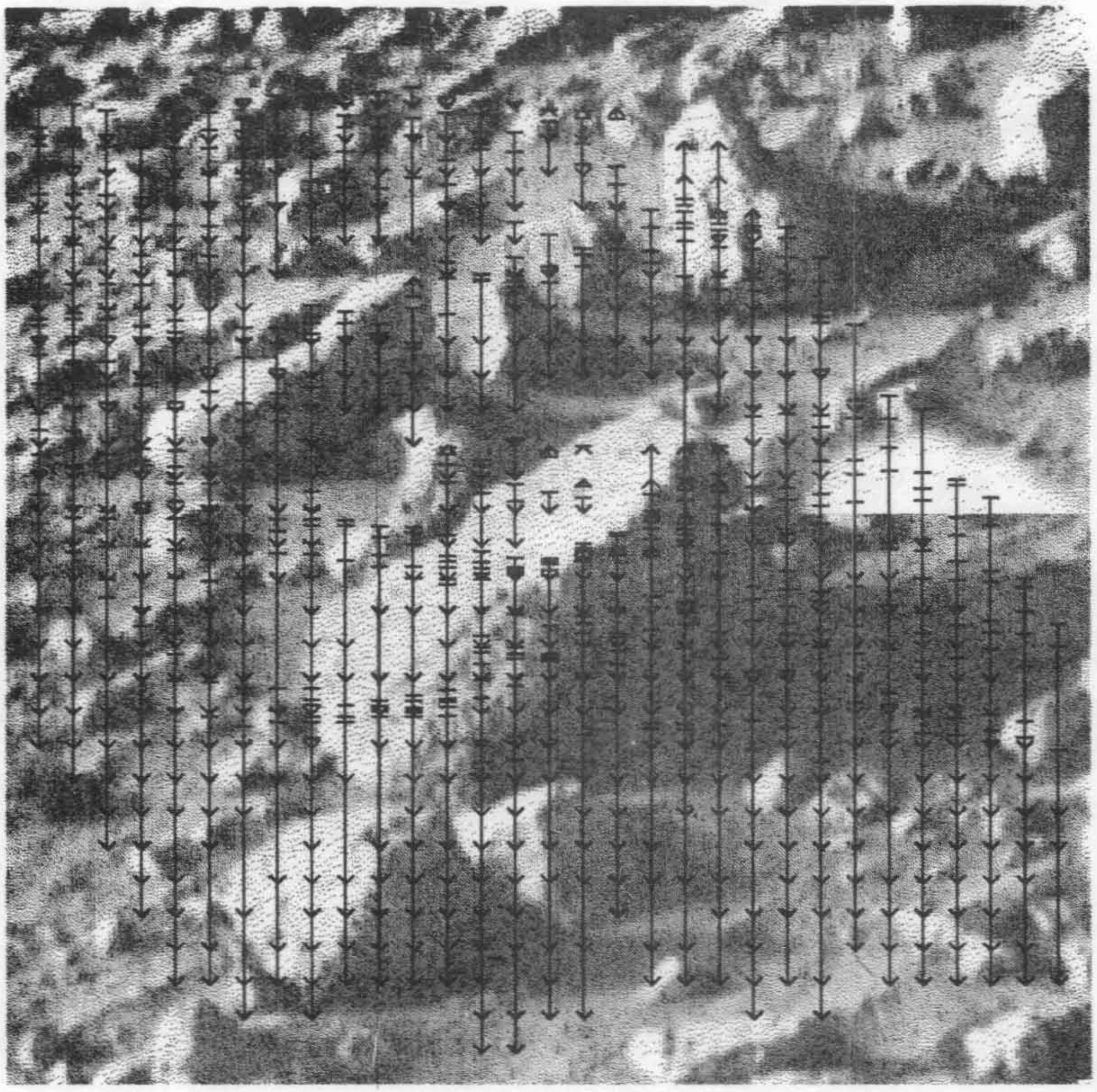

Figure 6-2. Points found by stereo processing, showing heights above reference plane.

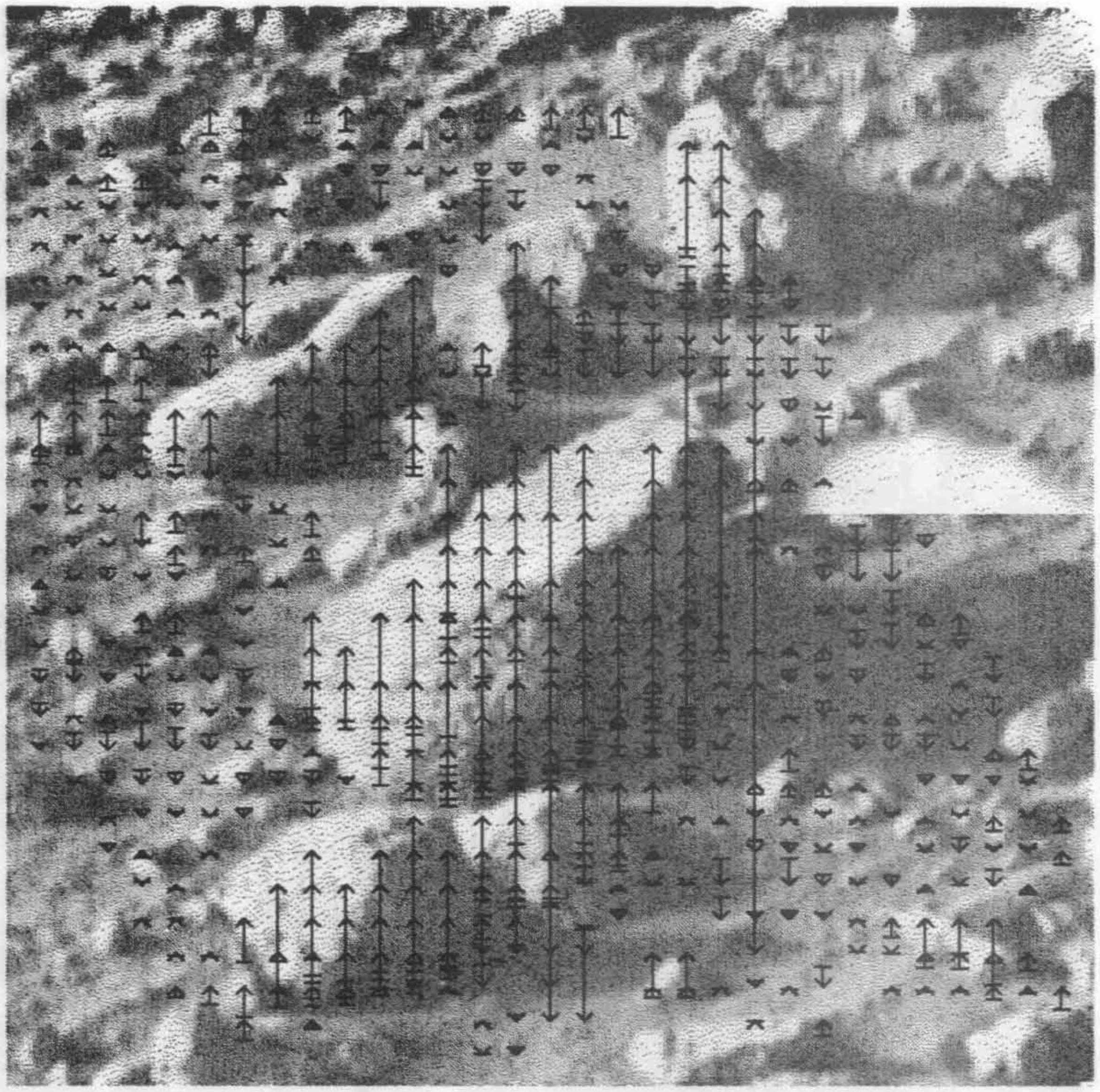

Figure 6-3. Heights above computed ground plane.

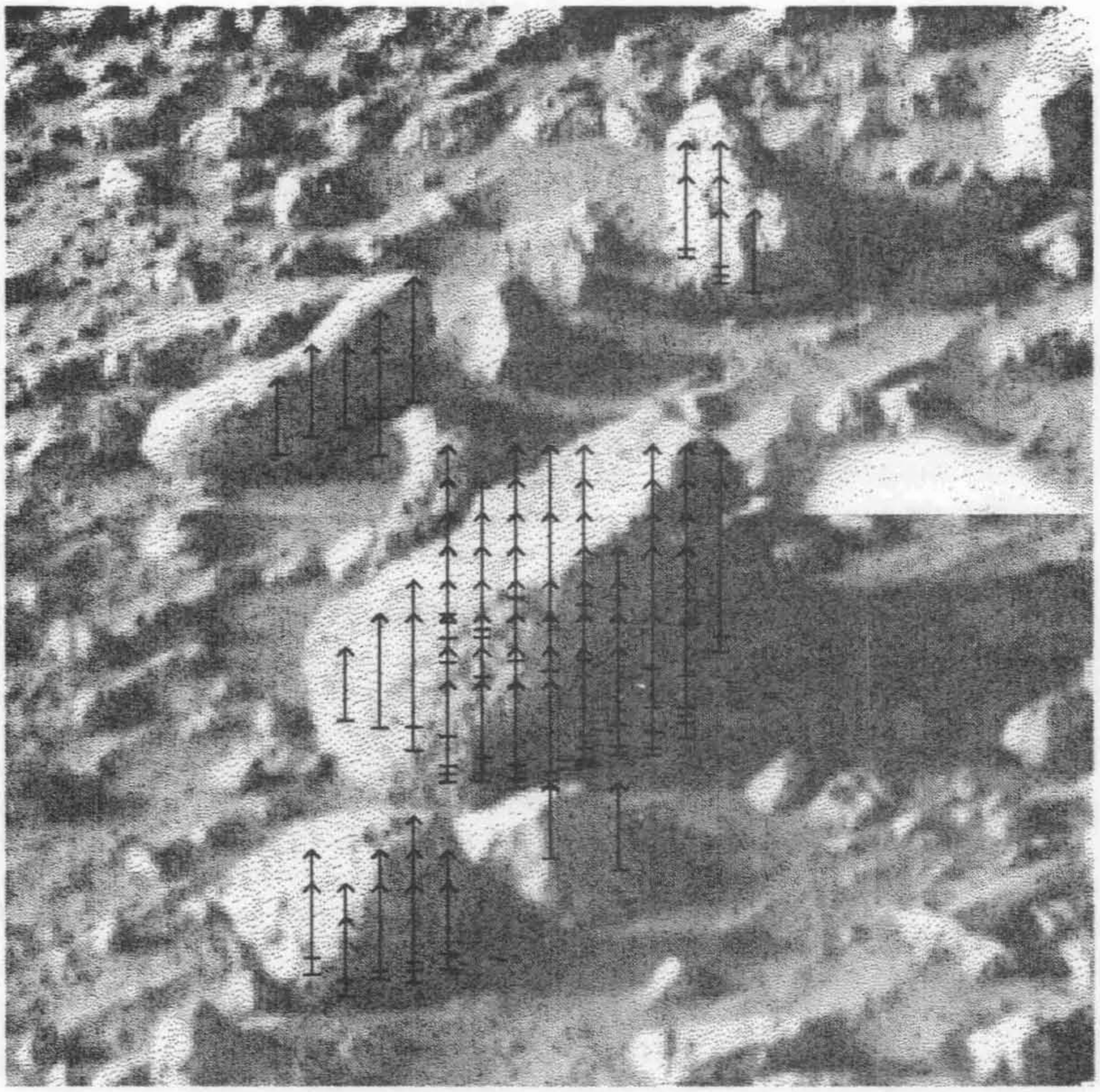

Figure 6-4. Heights above computed ground plane, for points above 5 cm.

needed, and equation (A.1-23) can be used directly to obtain each fit needed in the above algorithm, with the following substitutions:

$$P = [1 \; x \; y \; x^2 \; xy \; y^2]$$

$$E = [z] \tag{6.1-3}$$

$$D = [a \; b \; c \; d \; e \; f]^T$$

where z is the vertical coordinate, and where x, y, z, P, and E are evaluated for each point used in this fit (corresponding to the subscript i in equation (A.1-23)).

Figure 6-1 shows a ground surface (constrained to be a plane) fit to the data from the Mars pictures shown in Chapter 4. This is a horizontal view in a direction chosen so that the plane projects as a line. The vertical scale is exaggerated. Each point that was used in the final accepted fit is shown as a solid circle. The rejected points are shown as open circles. The number 1 in the upper left corner indicates that this was the first fit found according to step 3, the asterisk indicates that this was accepted as the final solution, the number 0.1503056 is the score for this solution, and the arrow in the upper right corner indicates the direction of the view in the horizontal plane (in this case about 45^0 to the left of the plane perpendicular to the baseline connecting the two cameras).

Figures 6-2 and 6-3 show the same data projected into the left picture in a "before" and "after" presentation. The head of each arrow is at one of the points used (corresponding to the dots in Figure 4-1), and the base of the arrow is on a reference plane 1.3 meters below the camera in Figure 6-2 or on the computed ground plane in Figure 6-3. The arrows are perpendicular to the reference plane in either case. The fact that most of the arrows point down in Figure 6-2 indicates that the ground is below the reference plane. The fact that most of the arrows in Figure 6-3 are very short where there are no large rocks indicates that a reasonable fit to the ground was obtained. Figure 6-4 is the same as Figure 6-3 except that only points at least five centimeters above the computed ground plane are shown.

6.2 Extension to Large Areas

If it is desired to fit a large ground area, a single paraboloid may not be a reasonable approximation, and some modification of the method in the previous section is needed. However, whatever method is used, some assumption about the smoothness of the ground is needed. Otherwise, the distinction between ground and objects disappears without more information other than the three-dimensional positions of points. Several possible approaches are discussed in this section. However, none of these have been implemented.

The simplest approach is to divide the area into small sections in a predetermined manner and to perform the solution of the previous section independently on each section, either with a plane or paraboloid fit. This will cause discontinuities in the ground to appear at the boundaries, but this effect is not troublesome if the only purpose of computing the ground is for thresholding heights for object detection. However, allowing these discontinuities to occur means that the constraint of smoothness has been disregarded at these boundaries, resulting in a less than optimum solution. This can be especially bad if some of the sections do not have enough points to determine the ground well. A possible refinement of this method is to take for each section not necessarily the best solution found, but the one which agrees best with the solutions for neighboring sections, if there are a few almost equally good solutions.

Some methods will now be discussed which utilize the constraint of smoothness and by means of a single fit produce a ground surface which varies smoothly but in a more or less arbitrary way over a large area. Because the surface in one part of the scene is almost independent of the surface in a distant part of the scene, the part of the algorithm which drives downward to eliminate clutter (steps 1, 2, 3, 7, and 8) cannot be used as part of this solution. Otherwise, a good solution in one part of the scene might be coupled with a bad solution in another part. Therefore, with these methods an initial approximation to the ground should be computed first using the method in the previous paragraph (separate solutions for predetermined sections) with fairly large sections, and then this solution should be refined using steps 4, 5, 6, and 9 on all of the data, with one of the methods discussed below.

One approach is to partition the area as above, but to perform a single solution which includes continuity constraints at the boundaries. For example, a tesselation into equilateral triangles can be used, with a separate plane fit in each triangle, but constrained so that the heights are continuous at the boundaries, whereas the derivatives may be discontinuous. (This works because three points determine a plane.) The coefficients to be computed could then be the heights at the corners of the triangles. (These would form the D vector.) The solution is linear in these heights, so apart from a different P matrix, the same method as in the previous section can be used. However, in order to produce a smooth surface, a priori weights would be used to minimize the change in slopes at the boundaries. Thus, the area can be divided into very small triangles, but this weight causes a smooth surface to be produced. The change in slope at a boundary is proportional to $h_1-h_2-h_3+h_4$, where h_2 and h_3 are the heights of the surface at the two vertices on this boundary, and h_1 and h_4 are the two vertices at the opposite corners of these two triangles from this boundary. The elements of the P matrix for these a priori observations are then 1, -1, -1, and 1, respectively. Thus the terms of the P^TP matrix corresponding to h_1 and h_2, h_1 and h_3, h_4 and h_2, and h_4 and h_3 are all -1, and the term corresponding to h_1 and h_4, h_2 and h_3, and all four diagonal terms are all 1. These would be multiplied by an appropriate weight and added into the H matrix at the correct positions for these terms, in order to constrain the change in slope at this boundary to be close to zero. This would be done for every boundary. (A approach similar to this method is to represent the ground by means of splines, discussed by Schultz [1973].)

Suppose that it is desired that the ground surface be smooth in most places but be allowed to have occasional discontinuites in slope, which would correspond to such things as banks and cliffs in the scene. This might be accomplished by modifying the above method as follows. The weights on the zero values in change in slope would be a function of the change in slope: perhaps a large constant value up to a certain small value of slope and a small value beyond. However, this causes the problem to be nonlinear, and it might require an inner loop of iterations in addition to the iterations which determine which points to use. Furthermore, on early iterations the large weights should be fairly small and the threshold for changing weights should be fairly high. These would gradually increase and decrease respectively as the iterations progress. This process would cause the creases in the surface to occur at approximately the right places. The precise initial values used would determine at about what size threshold the distinction between discontinuities in the slope of the ground and objects which are above the ground is made. It would also be possible to make the position of the vertices at which these discontinuites occur (in addition to their heights) parameters to be adjusted in the solution, but this would introduce even more nonlinearities. (Note that even without this additional adjustment, which vertices get the discontinuites is variable, but the solution is limited to the arbitrarily predetermined position of vertices.) It is not clear how well this would really work.

Another approach is to let the ground surface be the sum of a set of overlapping two-dimensional Gaussian functions (normal curves). The width (standard deviation) of these functions would be predetermined according to the desired smoothness of the ground surface, and the positions of the centers of the functions would be at a set of equally spaced points covering the area to be fit. The spacing would be sufficiently small so that insignificant ripple would be produced by the finite spacing. The parameters to be adjusted would then be the amplitudes of the Gaussian functions. The Gaussian function is chosen because of its smoothness and the rapidity with which it approaches zero in both directions. In some rough sense it has the optimum combination of these properties.

In order to keep the amount of computing within reasonable bounds, instead of using the actual Gaussian function (which extends to infinity in both directions), an approximation to the Gaussian function obtained by truncating it at a finite span would be preferred. Three or four standard deviations in each direction is a reasonable choice, since the value of the function at these points is only 0.011 or 0.00034 of its peak value. Furthermore, the approximation can be improved by subtracting the value of the Gaussian function at the truncation point from all of the values, in order to remove the discontinuity from the approximation function. (This function has been used previously in digital filters, for similar reasons, by Gennery [1966].)

Therefore, the equation for the ground surface would be

$$h = \sum_i a_i \omega_i \tag{6.2-1}$$

where

$$\omega_i = \exp\left(-\frac{(x-x_i)^2 + (y-y_i)^2}{2\sigma^2}\right) - \exp\left(-\frac{r^2}{2}\right), \quad \text{if } (x-x_i)^2 + (y-y_i)^2 < r^2$$

$$\omega_i = 0, \quad \text{otherwise}$$

where r is the number of standard deviations at which to truncate the Gaussian function. The quantities x_i and y_i are the centers of the Gaussian functions and are constant. The only quantities to adjust are the coefficients a_i. Thus the problem is linear. The elements of the P matrix in Appendix A for each data point would just be the ω_i quantities above.

There should be included in the solution a small amount of weight on the equality of adjacent a_i's, so that the surface will continue with a reasonable interpolation through areas where there are not many points being used to determine the surface. (This is also desirable to prevent the H matrix from becoming nearly singular if the spacing of the functions is small compared to σ.) This is done by adding 1 on the main diagonal of the H matrix at the position corresponding to each a_i of an adjacent pair, and -1 at the two off-diagonal positions corresponding to these two terms, all times the appropriate weight. This is done for all adjacent pairs. A large weight should not be used here, for this would introduce additional smoothing in the computed surface, and, if this is what is wanted, it would be more efficient to increase the width of the Gaussian functions (σ) and their spacing, and thus to decrease their number.

In order to decide what spacing to use for the ω_i's, equation (6.2-1) can be used with all $a_i = 1$, to see how much variation is produced in the values of h as a function of x and y with a given spacing. For example, if $r = \infty$ and the centers are on a square grid with spacing 2σ, the maximum ripple relative to the mean value is about 0.03; with spacing σ it is only about 10^{-8}. (With finite values of r, the former value would not change much unless $r < 3$, but in order to achieve a value as small as the latter would require a larger value of r.) A ripple of around one percent is probably tolerable unless the heights being fit are very large, in which case this would represent a large absolute error. In such a case the mean and perhaps the trend could be removed from the data first before it is used in the above method in order to reduce the size of the quantities being handled, and then the corresponding values would be added to the results. (This could be done by using the single-fit ground finder described in the previous section on the original data.)

It should be noticed that in both of the methods described above (plane triangles fit with smoothness constraints, and overlapping Gaussian functions) the H matrix which must be computed increases in size with increasing ground area covered, and it thus may be quite large. However, in both methods this matrix will be rather sparse. In the Gaussian case the matrix will be less sparse than in the other case (because the functions overlap), but because the spacing of the functions can be so large relative to the amount of smoothing produced, the size of the matrix can be considerably smaller than in the plane triangle case.

Of the two methods, the Gaussian method would appear to be superior because of the extremely smooth surfaces that can so easily be produced. However, there is no obvious way of adding the ability to fit discontinuities in slope, as can be done with the other method.

Chapter 7

OBJECT FINDER

This chapter describes the use of three-dimensional data for the detection of objects and the measurement of their position, size, and approximate shape. Although the three-dimensional data could be obtained from a scanning laser rangefinder, the object detector is designed to be tolerant of errors in this data, such as mistakes produced by incorrect matches in stereo vision data and poor accuracy of distances from stereo.

7.1 General Description

Many approaches are possible in describing the shapes of objects. At the extreme of simplicity each object could be represented by a sphere. Since a sphere can be specified by four parameters, this is economical, and, if the sphere encloses the actual object, it could suffice for obstacle avoidance, in a conservative way. Furthermore, this crude sort of information for each object in a large scene containing many objects amounts to fairly detailed information concerning the whole scene, and thus it would be useful for navigation. On the other hand, more elaborate descriptions that represent the object in more detail could be used. One possibility is the use of generalized cylinders or generalized cones, as by Nevatia and Binford [1977]. (In the simplest case, the generalized cylinder would reduce to an ordinary cylinder, which can be represented by seven parameters.) For man-made objects of regular form or elongated objects with well-defined axes, such a representation is very useful. However, for irregular objects such as rocks, the choice of how many parameters to use to describe the object and even the choice of direction of the axis of the generalized cylinder or cone may become almost indeterminate and thus may be greatly influenced by noise in the data. This would make the comparison of two object descriptions difficult.

A sort of compromise approach is used here, in which objects are represented by ellipsoids. Since objects can be approximated more closely this way than by spheres, in obstacle avoidance the vehicle may be able to pass more closely to the objects, and in navigation the shape information may aid in recognition of a scene. This is done at the cost of using nine parameters to describe an ellipsoid instead of four for a sphere, but the convenient mathematical properties of the sphere are mostly retained. The nine parameters could be the three coordinates of the center, three angles defining the orientation, and the semi-lengths of the three principal axes to define the size and shape. In this way the size and shape parameters would be independent of the choice of coordinate systems. However, for computational convenience the orientation, size, and shape are represented here by the six unique elements of a symmetrical 3-by-3 matrix (as in equation (7.3-1)), which are closely related to the second-degree coefficients in the general form of the equation of a quadric.

By "object" we do not necessarily mean here an actual physical object, but merely a portion of the scene that can be reasonably approximated by an ellipsoid. Thus, if we use as an example a vehicle exploring Mars, an object may be a single rock on the Martian surface, two or more adjacent rocks, or merely a bump in the ground. Also, an L-shaped physical object might be represented as two objects.

This ellipsoidal representation should be quite appropriate for representing rocks on Mars, because rocks probably tend to resemble more nearly ellipsoids than any other simple shape. However, it could also be used to represent cars in a parking lot or trees in a field, for example, especially in aerial photographs where the resolution may be poor compared to the size of the objects, and in other cases where precise object description or recognition is not necessary but rather an overall description of the scene is desired.

The stereo vision processing or laser rangefinder results in data representing the three-dimensional position of a large number of points distributed over the scene. The first step in the processing of this three-dimensional data is to find the ground surface, as described in Chapter 6. Then points which are above the ground by a sufficient amount (depending on the computed accuracy of the points, the roughness of the ground, and the minimum size of object that is of interest) are candidates for points on objects.

These above-ground points are clustered to produce preliminary groupings of points which correspond roughly to objects. An ellipsoid is fit to each cluster by first computing an initial approximation based upon the moments of the points in the cluster and then iterating a weighted nonlinear least-squares adjustment to fit the ellipsoids to these points and to avoid obscuring other points. Then, according to the relative positions of the ellipsoids and points, clusters can be broken or merged, and the process repeats until the apparently best segmentation is found. Each of these steps will be described in the following sections.

The object detection and measurement process as described here uses only three-dimensional position information. Brightness information is discarded after the stereo processing. However, a more complete system would use both types of information. Perhaps an edge detector could be applied to the brightness data in the regions near the outlines of the ellipsoids in order to refine the boundaries of the objects, for example.

7.2 Preliminary Clustering

Once the ground surface has been determined, all points that are above this surface by more than a threshold are clustered to form an initial approximation to the segmentation of the scene into objects.

Various clustering techniques could be used here. One possibility is a relaxation method, such as Zucker's [1976]. However, at present, the clustering is done by using the minimal spanning tree of the points. (The minimal spanning tree is the tree connecting all of the points such that the sum of the edges is minimum.) This is computed by using the nearest neighbor algorithm, as described in Duda and Hart [1973]. (The length of the edges of the tree is defined here as the three-dimensional Euclidean distance between the points.) Then the tree is broken at every edge whose length is greater than twice the average length of the adjacent edges, as suggested by Duda and Hart [1973]. However, a minimum length for an edge to be broken (related to the resolution of the data) is specified, so that the method will not be overly sensitive to local fluctuations in the data. Also, a maximum can be specified, beyond which all edges are broken.

7.3 Initial Approximations to Ellipsoids

Since each ellipsoid will be fit to a cluster of points by an iterative process, an initial approximation is needed. A good approximation increases the likelihood of convergence, decreases the number of iterations required, and can be used as the result in case the iterations do not converge. This initial approximation is obtained from the three-dimensional moments, through the second order, of the points in the cluster.

An ellipsoid can be represented by the following matrix equation:

$$(\mathbf{r}-\mathbf{c})^T W(\mathbf{r}-\mathbf{c}) = 1 \tag{7.3-1}$$

where $\mathbf{r}$ is a vector of the three-dimensional rectangular coordinates of any point on the surface of the ellipsoid, $\mathbf{c}$ similarly is the position of the center of the ellipsoid, and W is a positive-definite symmetrical 3-by-3 matrix. (See, for example, Hohn [1973].) Let M denote the inverse of W. (The square roots of the eigenvalues of M are the lengths of the semi-axes of the ellipsoid.) The relationship between the computed moments and the matrices $\mathbf{c}$ and M depends on the distribution of points over the ellipsoid. If the points are distributed uniformly over the ellipsoid, the vector $\mathbf{c}$ consists of simply the normalized first moments of the points. The matrix of normalized second moments about $\mathbf{c}$ of the points is $\frac{1}{5}M$ if the points are distributed uniformly through the body of the ellipsoid, or $\frac{1}{3}M$ if the points are distributed uniformly over the surface of the ellipsoid. If we have viewed the object from all sides, we might have an approximation to the latter case. However, if we have viewed it from a single point, we will have points distributed nonuniformly over half of the surface. (Actually slightly less than half will be seen because of perspective. Also, in stereo vision, both cameras must see each point, so that with a single pair of cameras only the common area seen from both camera positions will appear. These two effects will be neglected below, however.)

We assume here that the object is seen from a single viewpoint by a raster scanning device which produces points distributed uniformly in the image plane. Such a device might be a scanning laser rangefinder or an area-based stereo system. Actually, because of missing points, the distribution will not be uniform. It would be possible to estimate the actual distribution by computing higher-order moments, but this might be overly sensitive to randomness in the distribution or an inadequate density of points, so it is not attempted here. As an approximation, we assume an orthogonal projection instead of a central projection. Let **s** denote the vector of normalized first moments (centroid) and M_s denote the matrix of second moments about **s** obtained with this distribution, and let **o** denote the position of the camera.

The relationship connecting **s** and M_s to **c** and M can be derived by first considering the case of a sphere of radius ρ. A little integration shows that in this case the eigenvalue of M_s corresponding to the eigenvector **o**–**c** is $\frac{1}{18}\rho^2$, the other two eigenvalues are both $\frac{1}{4}\rho^2$, and **s** is **c** plus $\frac{2}{3}\rho$ times the unit vector in the **o**–**c** direction. All three eigenvalues of M should be ρ^2 in this case. An ellipsoid can be considered to be a distorted sphere (using stretching and skew distortions). Thus the ellipsoid can be considered to be stretched in the various directions by the amount given by the square roots of the ratios of the above eigenvalues, but in computing the displacement of the center, instead of ρ the distance from **c** towards **o** to the ellipsoid surface must be used. Thus the displacement of the center is the vector $\frac{2}{3}(\mathbf{o}-\mathbf{c})$ divided by the scalar $\sqrt{(\mathbf{o}-\mathbf{c})^T W(\mathbf{o}-\mathbf{c})}$. Since the points represented by **c**, **s**, and **o** are colinear, **c** can be replaced by **s** without changing the value of this ratio. Also, W $(=M^{-1})$ can be replaced by $\frac{1}{18}M_s^{-1}$ in this expression, because of the stretching discussed above. Thus **c** can be computed from **s** by translating by this amount. To compute M, we can take 4 times M_s to account for the factor of 4 in two dimensions, but this leaves 14 out of the factor of 18 by which we need to stretch the moments in the direction toward the camera. This extra amount can be introduced by adding 14 times the moment produced by a fictitious point at the intersection of the **s**-to-**o** line and the surface of the ellipsoid corresponding to M_s. In order to keep the ellipsoid to a reasonable shape when there are not enough points to determine it well, M as obtained above is averaged with a scalar matrix whose diagonal elements are h (which represents a sphere of radius $\sqrt{h}$), with the average weighted so that the sphere represents four additional points in the moment computation. The value of h is determined so that it is the average of the two components of the second moments at right angles to **o**–**s**, but limited by the average of all three components (the three eigenvalues), including the effect of 14 times the effect of the fictitious point, as above, as an upper limit, and excluding this effect, as a lower limit. This avoids putting undue weight on the **o**–**s** dimension when the ellipsoid is long in this direction, since this dimension is less reliable because of the factor of 18 compression.

By combining the above information, the computation of the initial approximation can be expressed as follows:

$$
\begin{aligned}
\mathbf{s} &= \frac{1}{n}\sum \mathbf{p} \\
M_s &= \frac{1}{n}\sum (\mathbf{p}-\mathbf{s})(\mathbf{p}-\mathbf{s})^T \\
M_t &= 4M_s + \frac{14(\mathbf{o}-\mathbf{s})(\mathbf{o}-\mathbf{s})^T}{(\mathbf{o}-\mathbf{s})^T M_s^{-1}(\mathbf{o}-\mathbf{s})} \\
\mathbf{c} &= \mathbf{s} - \frac{2\sqrt{2}(\mathbf{o}-\mathbf{s})}{\sqrt{(\mathbf{o}-\mathbf{s})^T M_s^{-1}(\mathbf{o}-\mathbf{s})}} \\
h &= \min\Big[\max\Big(\frac{4}{3}\operatorname{tr}(M_s), \\
&\qquad 2\operatorname{tr}(M_s) - 2\frac{(\mathbf{o}-\mathbf{s})^T M_s(\mathbf{o}-\mathbf{s})}{(\mathbf{o}-\mathbf{s})^T(\mathbf{o}-\mathbf{s})}\Big), \\
&\qquad \frac{1}{3}\operatorname{tr}(M_t)\Big] \\
M &= \frac{n}{n+4}M_t + \frac{4}{n+4}hI \\
W &= M^{-1}
\end{aligned}
\qquad (7.3\text{-}2)
$$

where $\mathbf{p}$ is the position of any point in the cluster, n is the number of points in the cluster, the summations are over these points, and I is the 3-by-3 identity matrix.

7.4 Iterative Solution for Ellipsoids

The adjustment of the ellipsoids is done by a modified least-squares approach. Each ellipsoid is adjusted so as to minimize the weighted sum of the squares of two kinds of discrepancies: the amounts by which the points (usually points in the cluster being fit) miss lying in the surface of the ellipsoid, and the amounts by which the ellipsoid hides any points as seen from the camera position. (In the latter case, the discrepancies actually should be considered separately for each camera that sees the point in question. However, for narrow-angle stereo we use as a reasonable approximation the assumption that the "camera" is at the midpoint of the stereo baseline.) Including the second kind of discrepancy is useful in helping to determine the size and shape of the object when the points on the object itself do not contain sufficient information. Also included in the weighted sum of squares to be minimized are a priori terms which tend to force the ellipsoid by default to become a sphere near the ground when the points do not constrain it well.

The first kind of discrepancy above optimally should be defined as the length of the normal from the point in question to the surface of the ellipsoid. However, computing this requires solving a sixth-degree equation. Therefore, as an expedient the distance between the point and the surface along a straight line from the center of the ellipsoid to the point is used instead. In order to be consistent with this definition, the second kind of discrepancy is defined as follows. The midpoint of the two intersections of the surface of the ellipsoid with a line from the camera to the point is first found. Then the discrepancy of the first kind is computed for this midpoint. (Note that if the main source of departure of the points from true ellipsoids is error in the measured position of the points, this is not the proper definition to use for the second kind. The normal distance from the point to the cone tangent to the ellipsoid with its vertex at the camera would be better. However, we assume that the major source of departure is the fact that the objects are not really ellipsoids, and thus the adopted definition is appropriate, because it is a measure of how far the ellipsoid juts out into the line of sight to the point.) Both kinds of discrepancies are illustrated in Figures 7-1 and 7-2.

Now we must consider exactly for which points which kind of discrepancy is computed. There are five regions of space to consider, according to whether the point is to the side of the ellipsoid as seen from the camera (that is, the line through the camera position and the point does not intersect the ellipsoid), is in front of the ellipsoid as seen from the camera, is inside the front portion of the ellipsoid (in front of the surface of midpoints as defined above), is inside the back portion of the ellipsoid, or is behind the ellipsoid. Also, there are two kinds of points to consider, according to whether or not the point is in the cluster which is assumed to correspond to this object. This produces ten combinations in all, which are illustrated in Figures 7-1 and 7-2. They divide into four categories.

First, if the point is not in the cluster and is either in front of the ellipsoid or is to the side, there is no discrepancy and this point is not included in the computations.

Second, if the point is in the cluster and is either in front, inside the front half, or to the side, or if the point is not in the cluster and is inside the front half, the first kind of discrepancy is used.

Third, if the point is not in the cluster and is behind the ellipsoid, or if either kind of point is inside the back half, the second kind of discrepancy is used.

Fourth, if the point is in the cluster and is behind the the ellipsoid, both kinds of discrepancies are used, and the point acts as two points in the computations. This is because there are two separate components of error in this case: the object does not extend enough on the side to hide the point, but it apparently bulges out in back (relative to the ellipsoid) to include the point.

In order to derive the mathematics for dividing space into the above five regions, consider the equation for the ellipsoid, as stated in (7.3-1), and the equation of a straight

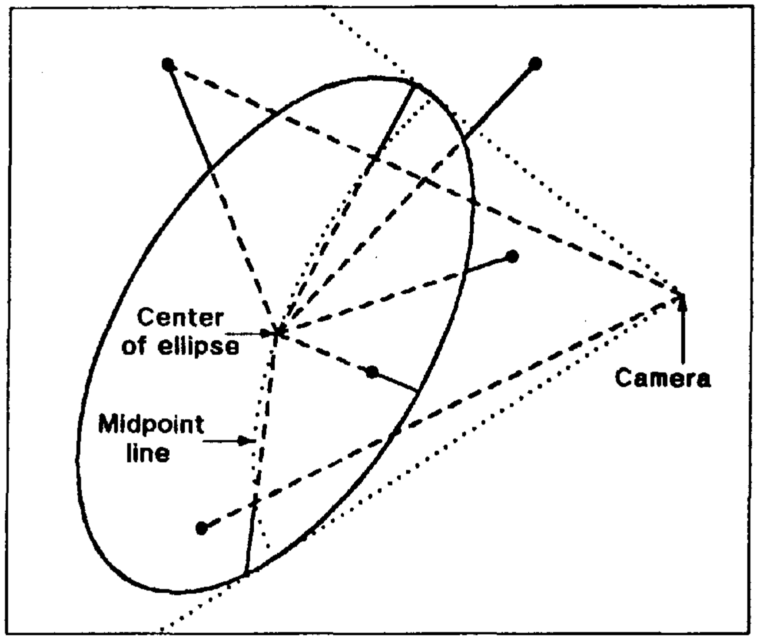

Figure 7-1. Two-dimensional version of adjustment, showing points belonging to this cluster. Solid dark straight lines are discrepancies to be minimized.

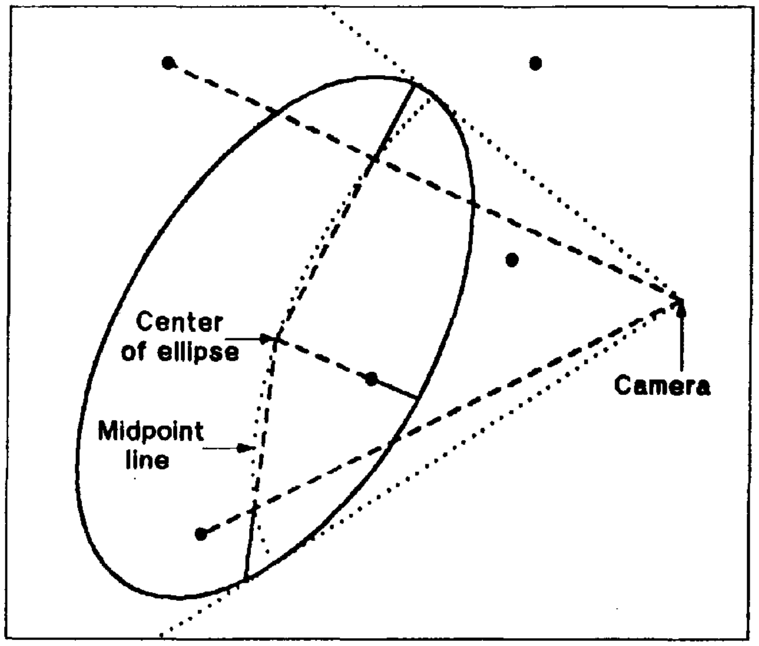

Figure 7-2. Two-dimensional version of adjustment, showing points not in this cluster. Solid dark straight lines are discrepancies to be minimized.

line through the camera position **o** and the point in question **p**, in parametric form,

$$(\mathbf{r}-\mathbf{o}) = u(\mathbf{p}-\mathbf{o}) \tag{7.4-1}$$

These can be combined to produce

$$[u(\mathbf{p}-\mathbf{o})+\mathbf{o}-\mathbf{c}]^{\mathrm{T}}W[u(\mathbf{p}-\mathbf{o})+\mathbf{o}-\mathbf{c}] = 1 \tag{7.4-2}$$

the roots of which determine the intersections of the line and ellipsoid. Equation (7.4-2) is equivalent to

$$\alpha u^2 + \beta u + \gamma = 0 \tag{7.4-3}$$

where

$$\alpha = (\mathbf{p}-\mathbf{o})^{\mathrm{T}}W(\mathbf{p}-\mathbf{o})$$

$$\beta = 2(\mathbf{p}-\mathbf{o})^{\mathrm{T}}W(\mathbf{o}-\mathbf{c})$$

$$\gamma = (\mathbf{o}-\mathbf{c})^{\mathrm{T}}W(\mathbf{o}-\mathbf{c}) - 1$$

The roots of equation (7.4-3) in u determine the region of space in which **p** lies. If the roots are imaginary ($\beta^2-4\alpha\gamma < 0$), the point is to the side of the ellipsoid. If the average of the two roots ($-\beta/2\alpha$) is positive, the point is in front of the midpoint surface, if negative, it is behind the surface. If the roots are real, the point is in front of, behind, or inside the ellipsoid according to whether both roots are greater than unity, both roots are less than unity, or unity lies between the roots, respectively. Alternatively, we can use the fact that the point is outside of the ellipsoid if and only if $(\mathbf{p}-\mathbf{c})^{\mathrm{T}}W(\mathbf{p}-\mathbf{c}) > 1$.

The discrepancy of the first kind is

$$\epsilon = \sqrt{(\mathbf{p}-\mathbf{c})^{\mathrm{T}}(\mathbf{p}-\mathbf{c})}\left(1 - \frac{1}{\sqrt{(\mathbf{p}-\mathbf{c})^{\mathrm{T}}W(\mathbf{p}-\mathbf{c})}}\right) \tag{7.4-4}$$

In order to compute the discrepancy of the second kind, the midpoint of the intersections of the camera-point line with the ellipsoid is first obtained as follows:

$$\mathbf{b} = -\frac{\beta}{2\alpha}(\mathbf{p}-\mathbf{o}) + \mathbf{o} \tag{7.4-5}$$

Then the discrepancy of the second kind is

$$\epsilon = \sqrt{(\mathbf{b}-\mathbf{c})^T(\mathbf{b}-\mathbf{c})}\left(1 - \frac{1}{\sqrt{(\mathbf{b}-\mathbf{c})^T W(\mathbf{b}-\mathbf{c})}}\right) \qquad (7.4\text{-}6)$$

Because there may be erroneous points in the data, points which have large discrepancies relative to the size of the ellipsoid are given less weight in the solution. The weighting function used is

$$\omega = \frac{1}{\frac{1 + 2\left(\sqrt{(\mathbf{r}-\mathbf{c})^T W(\mathbf{r}-\mathbf{c})} - 1\right)^2}{\mathrm{tr}(W)} + \sigma^2} \qquad (7.4\text{-}7)$$

where $\mathbf{r}$ represents $\mathbf{p}$ or $\mathbf{b}$ for discrepancies of the first or second kinds, respectively, and σ is the component of standard deviation of measurement errors in $\mathbf{p}$ propagated into the discrepancy. (If these are unknown, σ can be zero.) Thus the dimensionless quantity to be minimized (by adjusting $\mathbf{c}$ and W) is $\Sigma\omega\epsilon^2$, plus some additional terms for a priori values yet to be discussed. However, this quantity is minimized only with respect to the effects of $\mathbf{c}$ and W acting through ϵ and not their effects through ω.

In order to solve the above nonlinear problem, the Gauss method described in Appendix A is used. This method is equivalent to using the partial derivatives of the discrepancies to approximate the nonlinear problem by a linear statistical model, solving the linear problem, and iterating this process until it converges.

On any one iteration the following is done. The current values of $\mathbf{c}$ and W are used to compute for each point the value of ϵ as defined above and the 1-by-9 matrix P, which consists of the partial derivatives of ϵ with respect to the three elements of $\mathbf{c}$ and the six unique elements of W. (W is symmetrical.) The following summations over all of the points are computed, in which each point in the first category above is not used, each point in the second or third categories appears once, and each point in the fourth category appears twice:

$$\begin{aligned} H &= H_0 + \sum P^T\omega P \\ C &= C_0 + \sum P^T\omega\epsilon \end{aligned} \qquad (7.4\text{-}8)$$

(H_0 and C_0 are used for the a priori values yet to be discussed.) Then the 9-by-1 matrix of corrections is

$$D = vH^{-1}C \qquad (7.4\text{-}9)$$

where v is a factor used to improve convergence because of the very nonlinear nature of the problem. (Currently $v = 0.5$ on early iterations, but $v = 1$ after a test indicates that this will produce more rapid convergence.) The elements of D are subtracted from the corresponding elements of $\mathbf{c}$ and W to obtain the improved approximations for the next iteration. H^{-1} from the last iteration is the covariance matrix of the ellipsoid parameters, although it may need to be adjusted by a scale factor according to the size of the residuals of the final fit, as described in Appendix A.

Now the a priori values will be discussed. In some cases the points affecting the ellipsoid will be insufficient in number or insufficiently distributed to determine all parameters of the ellipsoid very well. It is therefore desirable to have a priori values for some of the parameters with appropriate weight in the solution to constrain them to reasonable default values when the points do not contain sufficient information. When there is ample information in the points, the a priori values will have very little effect because of their small weight. The a priori values currently used are the ground surface height directly under $\mathbf{c}$ for the vertical component of $\mathbf{c}$, with weight $0.1/\mathrm{tr}(M)$, equality for the diagonal elements of W, with weight $\mathrm{tr}(M)^2/10$, and zero for the off-diagonal elements of W, with weight $\mathrm{tr}(M)^2/10$, where $M = W^{-1}$. (Including $\mathrm{tr}(M)$ as shown scales things correctly so that the solution is invariant under a scale factor change.) The effect of the W terms is to try to force the ellipsoid into a spherical shape. (It would be better to apply the a priori weights to the principal semi-axes of the ellipsoid, trying to force them to equality, so that the effect of the a priori values would be independent of the coordinate system being used. This would require propagating these values into the elements of W on each iteration, so the implemented program uses the method described here instead.) These a priori terms are put into the solution in the following way. The diagonal element of H_0 corresponding to the vertical component of $\mathbf{c}$ is $0.1/\mathrm{tr}(M)$, the three diagonal elements corresponding to the off-diagonal elements of W are each $\mathrm{tr}(M)^2/10$, and the 3-by-3 submatrix on the diagonal of H_0 in the position corresponding to the diagonal elements of W consists of $\frac{2}{3}$ on its main diagonal and $-\frac{1}{3}$ elsewhere multiplied by $\mathrm{tr}(M)^2/10$. All other elements of the 9-by-9 matrix H_0 are zero. Then

$$C_0 = H_0 G \tag{7.4-10}$$

where G is a column matrix of the current values of $\mathbf{c}$ and W, arranged as in D, with the height of the ground directly under the center of the ellipsoid subtracted from the element of G corresponding to the vertical component of $\mathbf{c}$. H_0 and C_0 are used in the summations for H and C as previously shown.

7.5 Breaking and Merging Clusters

Because the preliminary clustering is dependent on local information, it may not produce the best segmentation based on more global information. Therefore, after ellipsoids have been fit to all of the preliminary clusters, these clusters may be tentatively broken into smaller clusters and merged into larger clusters, new ellipsoids are fit to these clusters by the same process previously described, and a decision on whether to keep or reject each of these actions is made based on the goodness of fit of the ellipsoids to the points.

In order to decide where to break a cluster, for each edge in the portion of the original minimal spanning tree which connects this cluster the quantity $\lambda(1-a)$ is computed, where λ is the length of the edge and a is the minimum of $\sqrt{(\mathbf{p}-\mathbf{c})^T W(\mathbf{p}-\mathbf{c})}$ for the two points connected by the edge. Then the cluster is tentatively broken at the edge for which this quantity is maximum, of all such edges such that each new cluster formed has at least four points at least one of which has $(\mathbf{p}-\mathbf{c})^T W(\mathbf{p}-\mathbf{c}) > 1$ (that is, it is outside the old ellipsoid). This process tends to break the cluster at places furthest inside the ellipsoid, but connecting points that are outside the ellipsoid. If this new clustering is accepted by the criteria described below, the process repeats on the new clusters.

After the above breaking process is finished, any two clusters are tentatively merged if $(\mathbf{c}'-\mathbf{c})^T W(\mathbf{c}'-\mathbf{c}) < 4$ for either cluster, where $\mathbf{c}'$ is $\mathbf{c}$ for the other cluster, provided that these two clusters were not previously one cluster before breaking. If there is competition for the merging, the cluster pair with the minimum value for this quantity is merged first. If a merger is accepted, further mergers can take place on these clusters by this same process.

The criteria for accepting two clusters or one after a tentative break or merger are as follows. If $(\mathbf{c}'-\mathbf{c})^T W(\mathbf{c}'-\mathbf{c}) < 1$ for either small cluster, where $\mathbf{c}'$ is $\mathbf{c}$ for the other small cluster (that is, the center of one ellipsoid is inside the other ellipsoid), the single cluster is chosen. Otherwise, the following quantity is computed for each of the three ellipsoids:

$$q = \frac{\Sigma\omega\epsilon^2}{n-3} + \frac{m}{10} \tag{7.5-1}$$

where ϵ and ω are the discrepancies and weights from the last iteration, as defined in the previous section, n is the number of points in the cluster corresponding to this ellipsoid, and m is the number of points below the height threshold but directly above the ellipsoid. If the initial approximation is used as the result, ϵ and ω are obtained from the first iteration, and the first denominator is n instead of $n-3$. (The second term, containing m, is included to penalize solutions which lie mostly below the ground, with the ellipsoid reaching above the ground in a small area to meet the points in its cluster.) Then the two small clusters are chosen if the sum of their two values of q is less than the value of q for the single cluster. Otherwise, the single cluster is chosen.

7.6 Example

Figure 7-3 shows the points in the Mars picture previously shown in Figures 6-2 and 6-3, but this time in a nominally vertical orthogonal projection (perpendicular to the reference plane). The figure covers an area 1.6 meters by 1.2 meters in the reference plane. The lower left corner is 0.6 meters to the right of the plane through the left camera and perpendicular to the baseline connecting the cameras, and it is 2.9 meters in front of the baseline connecting the cameras. The similar coordinates for the upper right corner are 1.8 meters and 4.5 meters. The symbol for each point represents height in centimeters above the computed ground plane, with the letters "A", "B", "C", etc. representing the values 10, 11, 12, etc.

A 5-centimeter height threshold was used for selecting the points to cluster in the object finder. The minimum distance for breaking the minimal spanning tree to form the initial clusters was also 5 centimeters, and the maximum distance for connecting points was 20 centimeters. (Using zero and infinity for this minimum and maximum distance produced an identical clustering in this case.)

Figure 7-4 shows the points that passed the height threshold. These points are connected to show the minimal spanning trees that were computed. Solid lines connect points within each initial cluster.

Figure 7-5 shows the ellipsoids that were fit to the initial clusters. Each ellipsoid is represented by two ellipses. One ellipse is the orthogonal projection of the ellipsoid onto the reference plane. The other ellipse is the intersection of the ellipsoid with a plane through the center of the ellipsoid and parallel to the reference plane. (In most cases the two ellipses almost coincide and thus cannot be distinguished in the figure.) Only the clustered points are shown here, as in Figure 7-4. However, as previously described, any of the points shown in Figure 7-3 may have been involved in the adjustment of the ellipsoids. Remember that the fit is done in three dimensions, whereas Figure 7-5 shows a two-dimensional projection.

Figure 7-6 shows in the same way the results of the breaking and merging operations. The two clusters in the center (corresponding to the large rock in the center of the pictures) were merged into one, and a new ellipsoid is shown for this cluster. The other clusters were not changed.

These results were projected into the left picture to produce Figure 7-7. The outline of the ellipsoids as they would be seen from the left camera are superimposed on the picture. The lengths of the principal axes of the large ellipsoid in the center are 30.8, 26.2, and 16.6 centimeters.

Notice that the ellipsoid fit to the rock in the upper right corner is much too large. This is because the only points found on this rock were on its fairly flat face, and no points were found in the background behind the rock to help to constrain its size, as can be seen in Figure 6-3. This lack of points was caused by the fact that most of the desired region is outside of the right picture, as can be seen in Figure 4-2. (In such a case the covariance

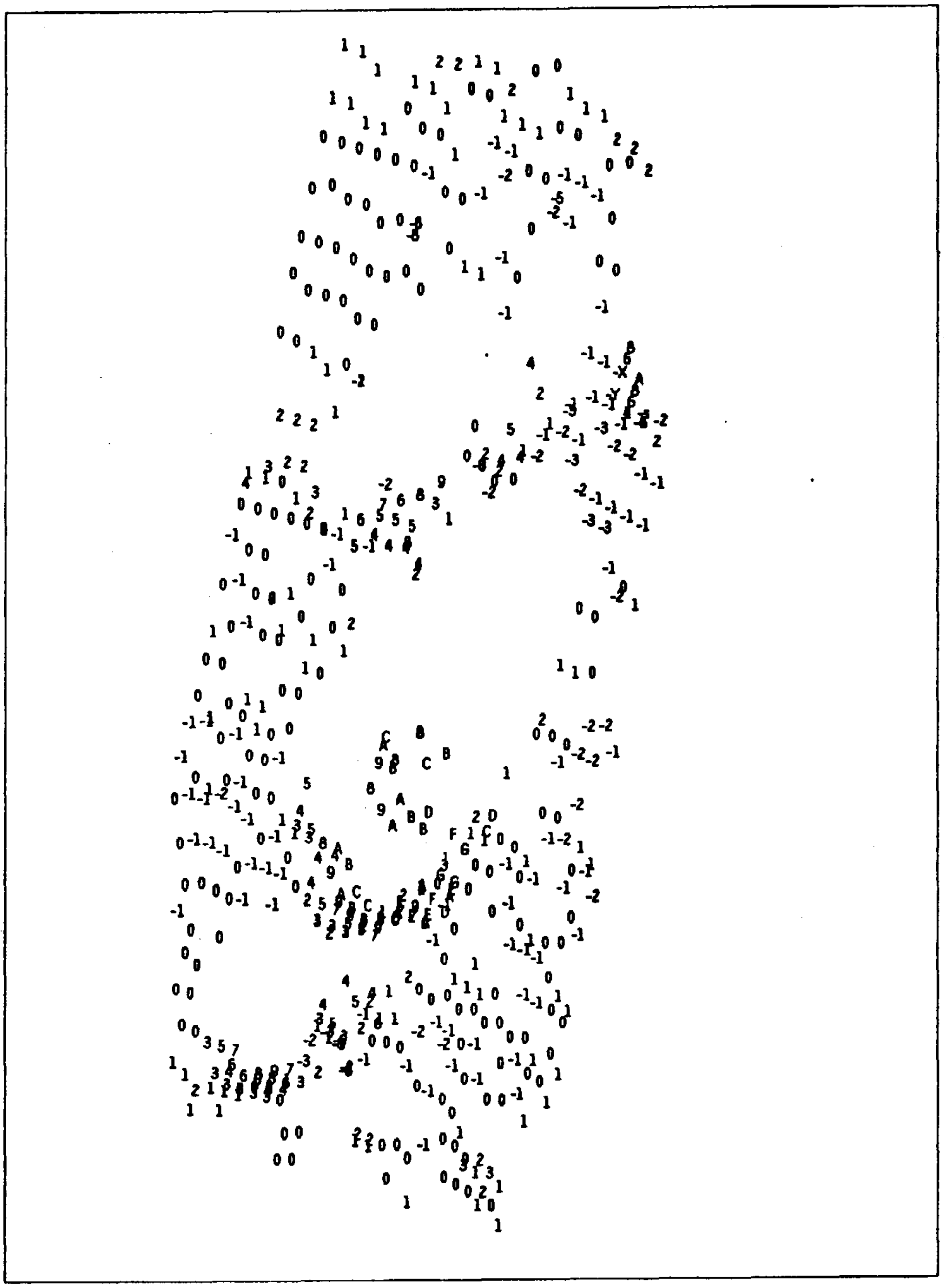

Figure 7-3. Vertical view of points, showing heights above computed ground plane in centimeters.

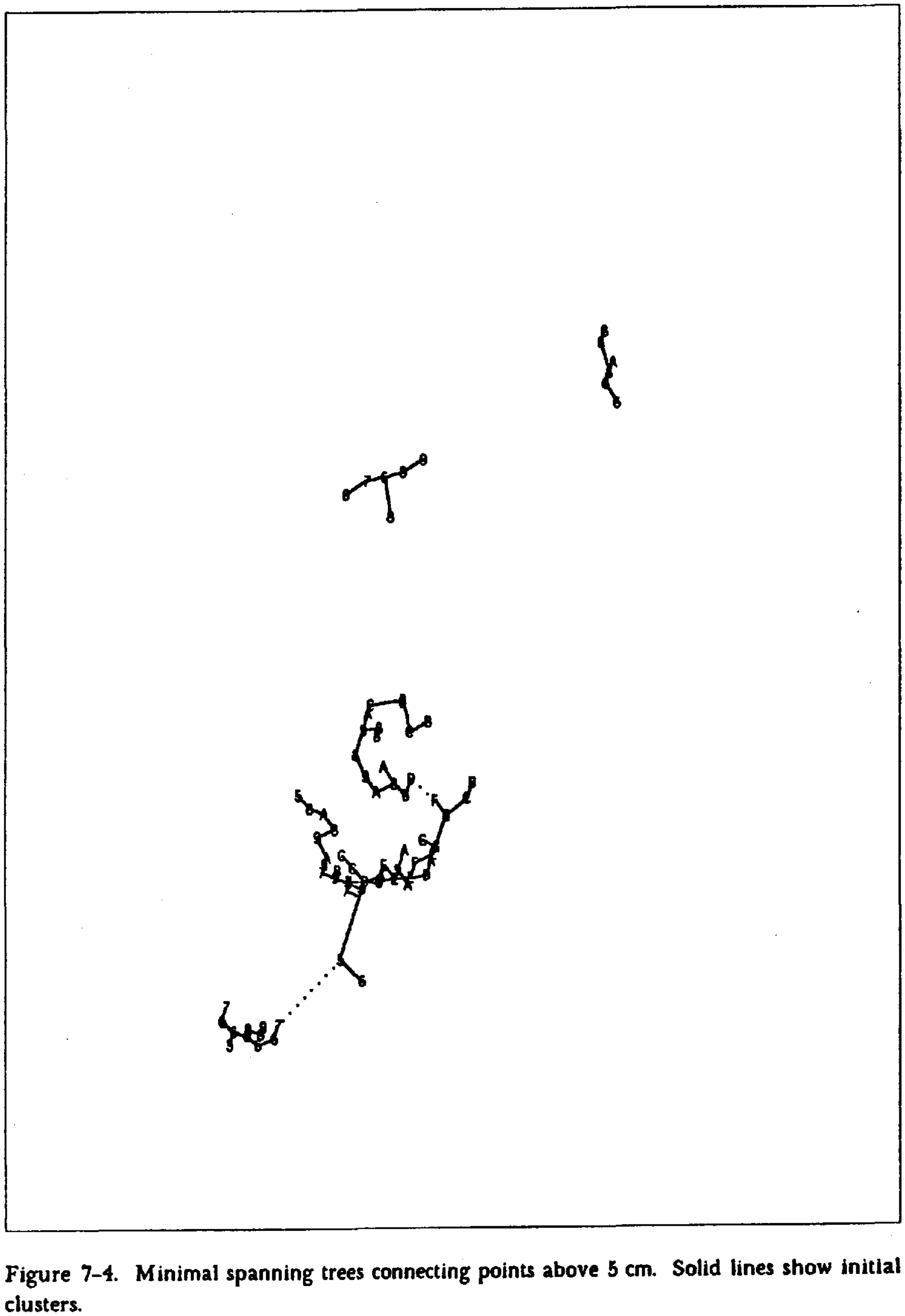

Figure 7-4. Minimal spanning trees connecting points above 5 cm. Solid lines show initial clusters.

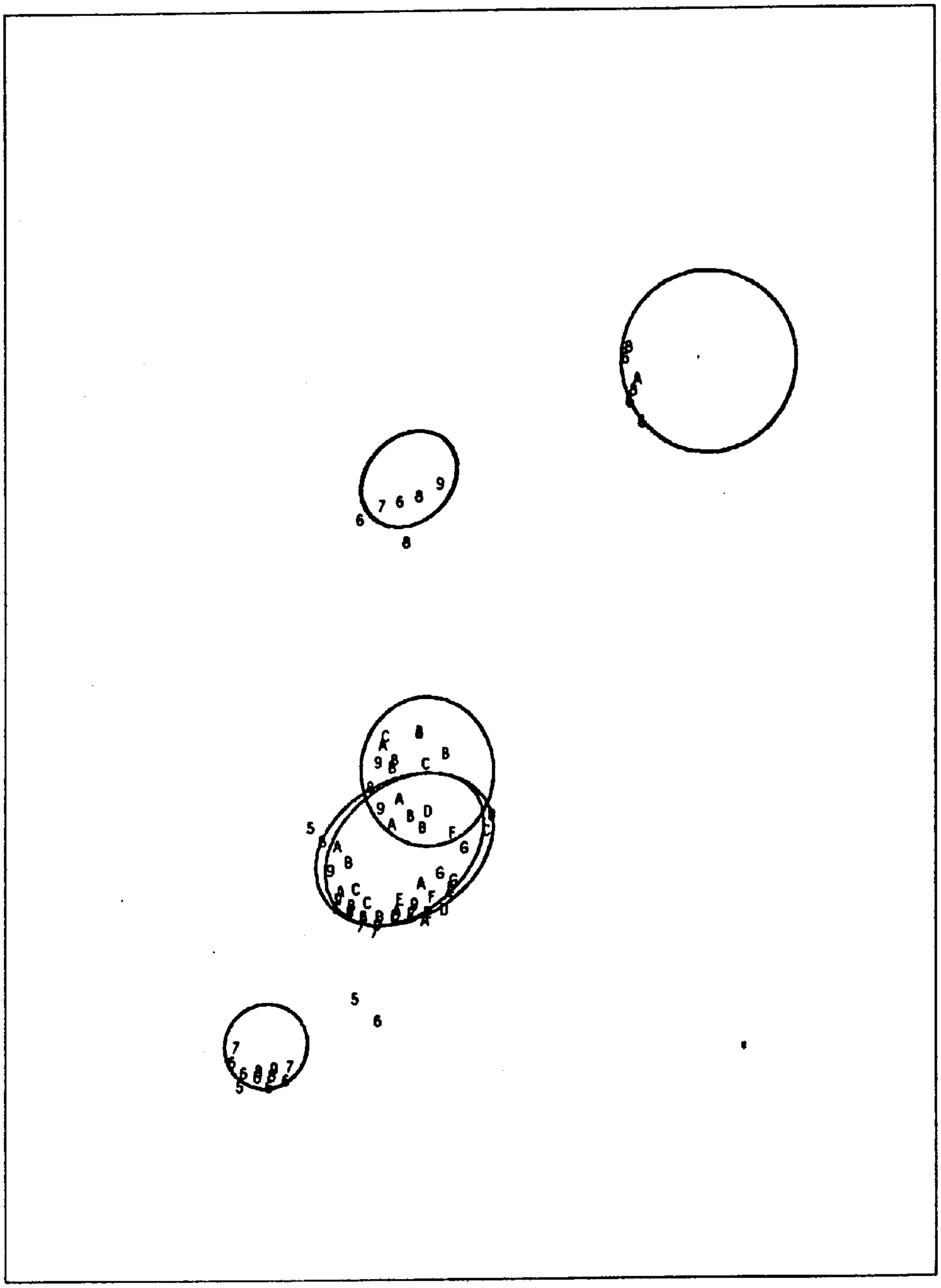

Figure 7-5. Ellipsoids fit to initial clusters.

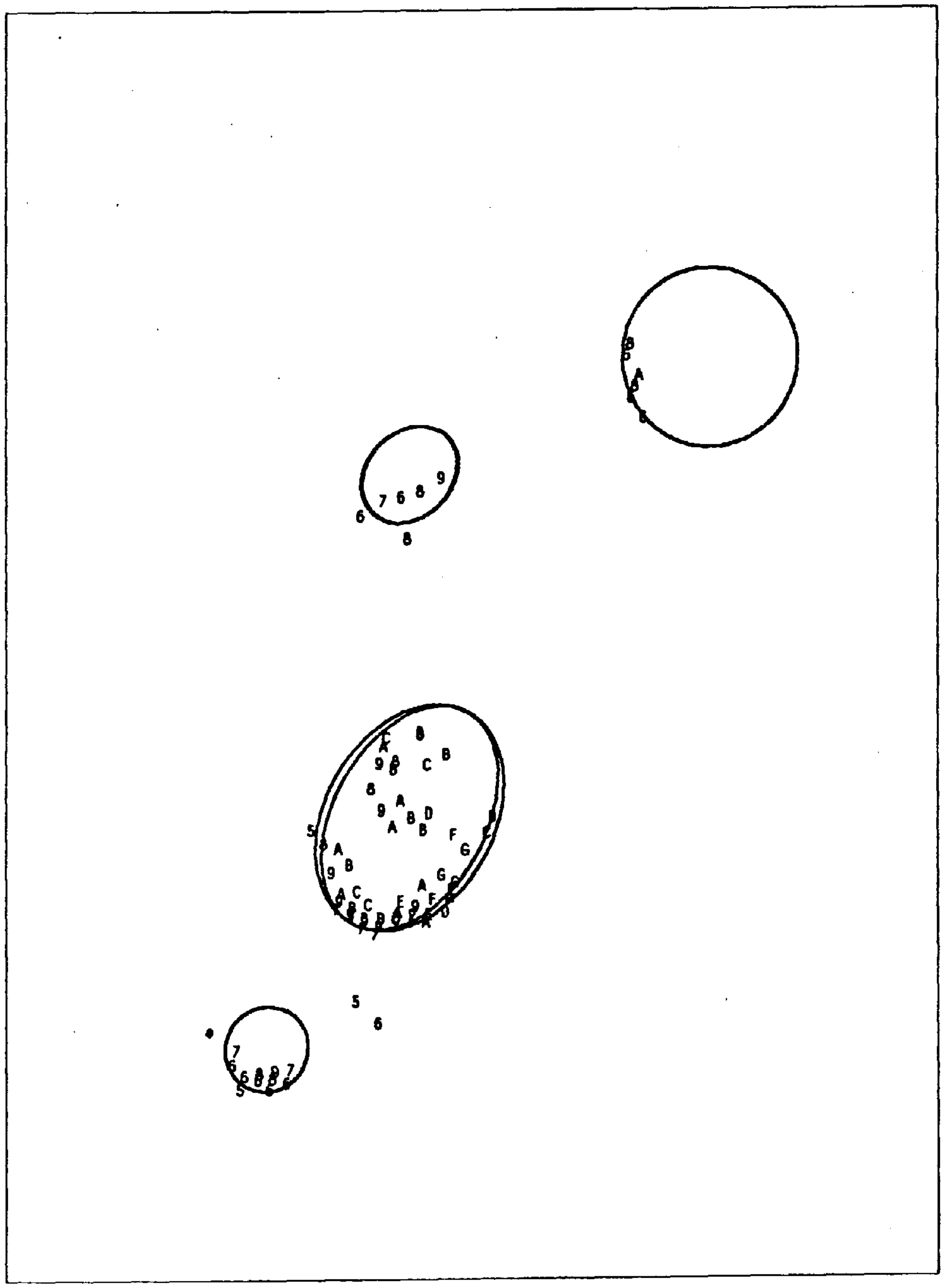

Figure 7-6. Ellipsoids fit to final clusters.

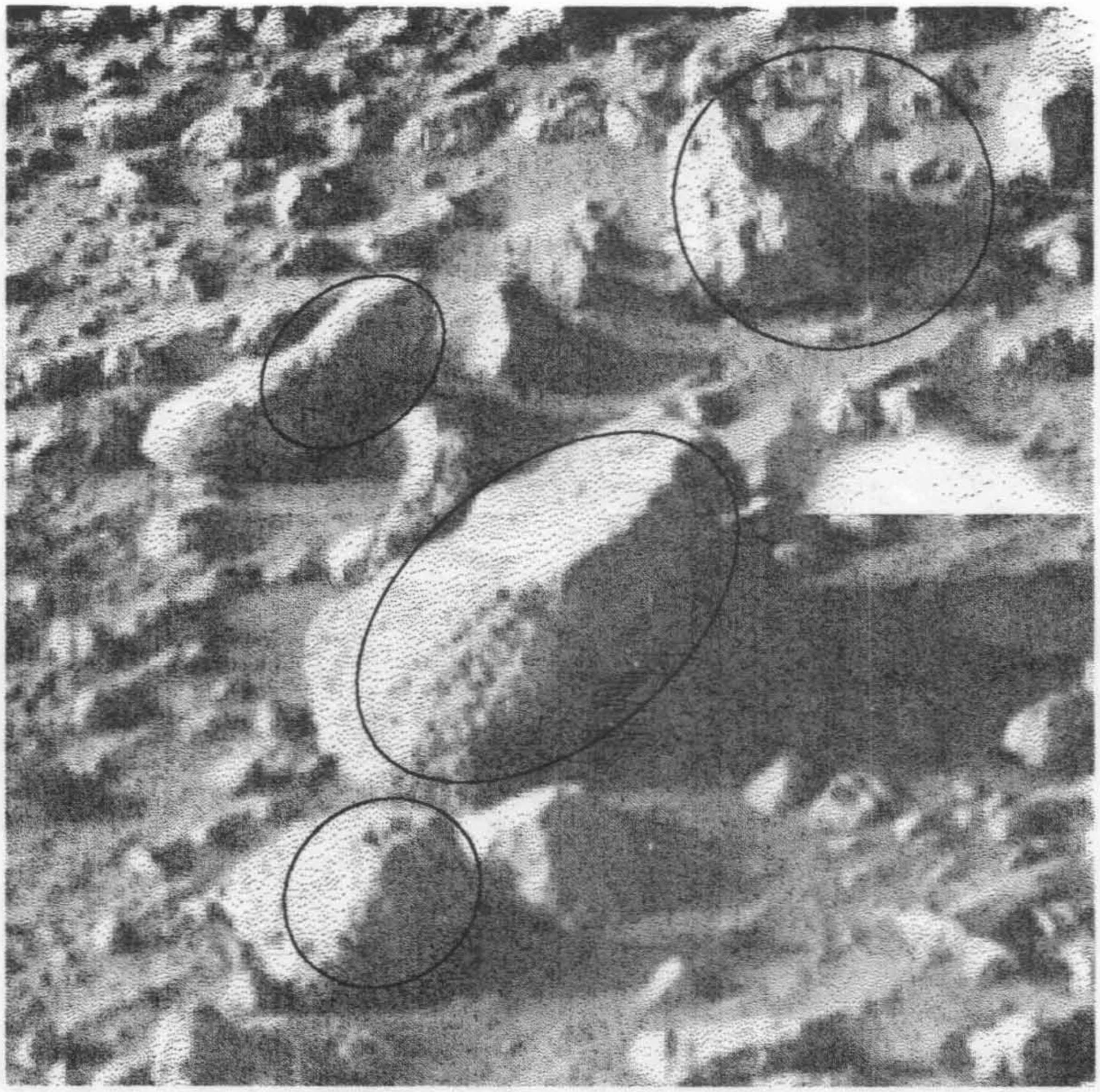

Figure 7-7. Ellipsoids fit to final clusters, projected into left picture.

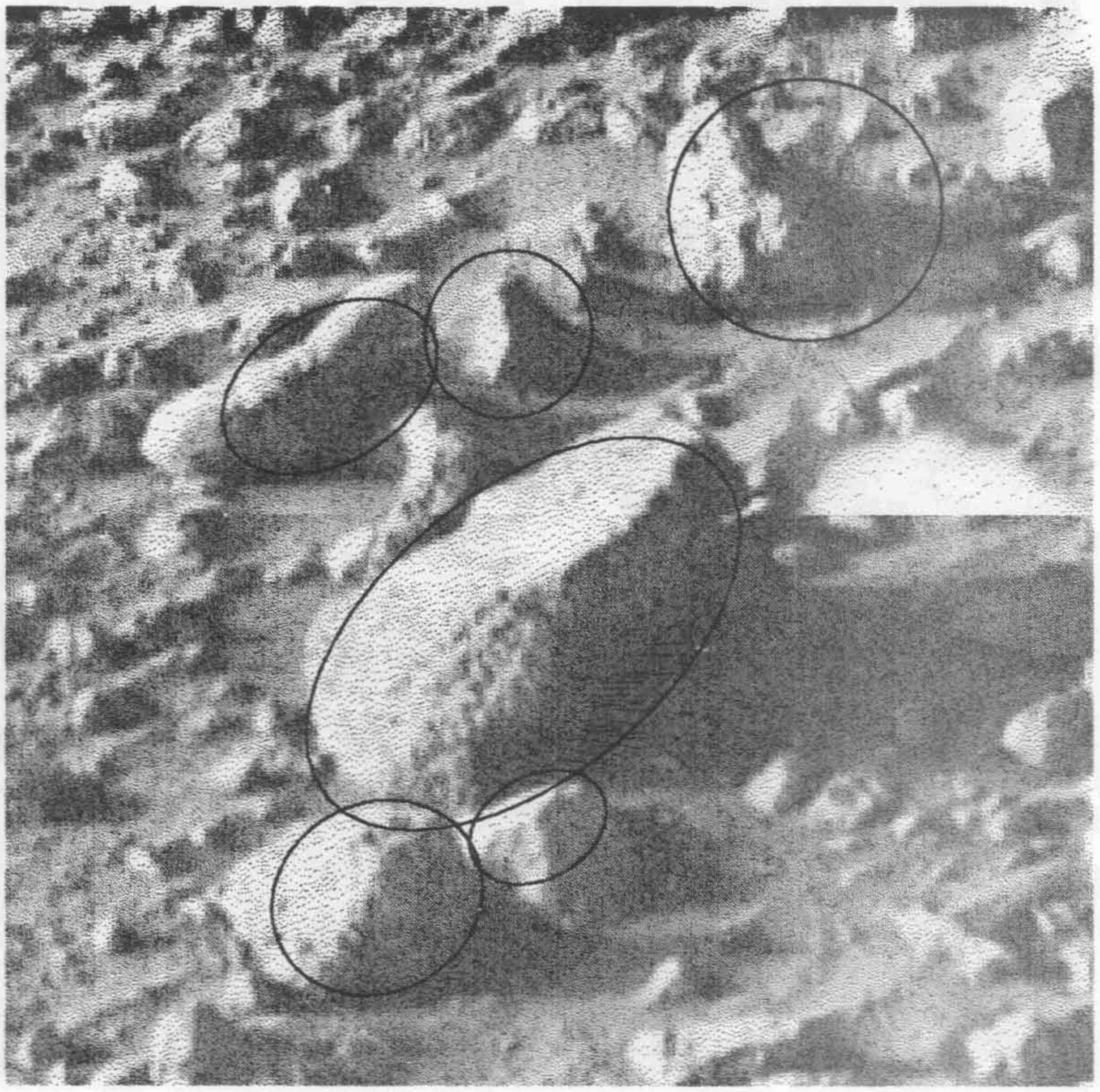

Figure 7-8. Results using 3-cm height threshold instead of 5 cm.

matrix of the ellipsoid parameters indicates the large uncertainty in its size and shape.)

Figure 7-8 shows the results of processing the data slightly differently. A height threshold of 3 centimeters instead of 5 centimeters was used. Some of the smaller rocks are detected in this case.

Chapter 8

MATCHING OF SCENES

The previous chapter described a means of modelling three-dimensional scenes in terms of ellipsoidal objects. A method of matching such scene descriptions will now be described. This method uses the covariance matrices generated by the object finder, which indicates the accuracy of the object parameters, to determine the goodness of match for each object match. This is a special case of a more general problem, in which there is a set of features in each scene, with each feature being represented by a vector of feature parameters and the covariance matrix of these parameters. The method to be described applies to this general case, but it will be described in terms of the special case at hand. In this special case, the vector of feature parameters consists of the parameters describing the position, size, and shape of an ellipsoid (the $\mathbf{c}$ vector and the W matrix in Chapter 7). There are nine of these parameters in the complete case (three elements of $\mathbf{c}$ and six elements of W). However, it may be desired to eliminate the vertical component of position (third element of $\mathbf{c}$), because this component is less reliable due to uncertainties in the vertical position of a roving vehicle, and this is done in the implemented version of the program, leaving eight parameters actually used. Also, a completely general program would allow for translations and rotations in three dimensions, whereas the method described below allows only translations and rotations in the horizontal plane. This is suitable for a roving vehicle, since it should be able to obtain an accurate vertical from gravity. (In fact, with reasonable instrumentation the direction in the horizontal plane should be determined also, leaving only translation to determine.)

8.1 Optimum Match

The problem at hand can now be described fully. Given are two scene descriptions. Scene 1 consists of n objects and Scene 2 consists of n' objects. Each object is described by a vector X and its covariance matrix S, with primes denoting objects in Scene 2. Also available for each object is a probability b that this object will be present in the other scene, if the two scenes actually refer to the same physical scene. (These probabilities could be estimated from statistics gathered from experience with the system that produced the data, and might be a function of the size of the object, since the larger objects would be more likely to be detected in the other scene and would be less likely to be spurious.) Other general information that is available includes the a priori probability p_0 that the two scenes match (that is, refer to the same physical scene), the a priori value of the rotation θ_0 of Scene 2 relative to Scene 1 and its standard deviation σ_{θ_0}, and the a priori value of the scale factor f_0 of Scene 2 relative to Scene 1 and its standard deviation σ_{f_0}. It is desired to find the translation Δx and Δy of Scene 2 relative to Scene 1, the rotation θ, the scale factor f, the standard deviations of these quantities, and the probability p that the two scenes actually match.

The approach used here uses Bayes' theorem, which in general states the following:

$$\tilde{p}_i = \frac{\pi_i p_i}{\sum_i \pi_i p_i} \qquad (8.1\text{-}1)$$

where π_i is the a priori probability of event i occurring, p_i is the probability that the observed result would occur given that event i occurs, and $\tilde{p}_i$ is the a posteriori probability that event i has occurred, given that the observed result has occurred.

For the present purposes, an event will be considered to be the fact that the two scenes match and the occurrence of a particular set of matches between the objects in Scene 1 and the objects in Scene 2. The above terminology is altered slightly to include the fact that the scenes actually match in these events, with an extra event being the scenes not matching. Then Bayes' theorem can be restated as follows:

$$p_k = \frac{p_0 \pi_k \rho_k}{p_0 \sum_k \pi_k \rho_k + (1-p_0)\rho_0} \qquad (8.1\text{-}2)$$

where p_0 is the a priori probability that the scenes match, as previously defined, π_k is now the a priori probability that the kth combination of object matches would occur, given that the scenes match, ρ_k is the probability density of the observed set of object parameters occurring, given that the kth match is correct, ρ_0 is the a priori probability density of the observed set of object parameters, and p_k is the a posteriori probability of the kth set of matches being correct. Thus the term $p_0\pi_k$ is the a priori probability of the kth event, and the term $(1-p_0)$ is the a priori probability of the scenes not matching.

If the combination of object matches for which p_k is maximum is found, and if this value of p_k is large (near unity), then this combination can be assumed to be correct, and it can be used to determine the desired parameters describing the translation, rotation, and scale factor by a process to be described. (It would appear that all combinations of object matches would have to be used in this computation, but a way of avoiding this will be described in the next section.) However, the computation of π_k and ρ_k needed in (8.1-2) will be described first.

The π_k quantities can be found by the following reasoning. The probability that object i in scene 1 will be matched to some object in Scene 2 is b_i, the probability that it will be unmatched is $1-b_i$, and similarly for b_j' and $1-b_j'$ for object j in Scene 2. Thus the probability of a particular subset of m objects from Scene 1 and m objects from Scene 2 being matched, and no others, is the product of these terms over all objects, chosen according to whether each object is matched or not. However, there are $m!$ ways of matching a set of m objects to another set of m objects. Therefore, this product is divided by $m!$ to obtain the a priori probability of an individual matching combination. Thus,

$$\pi_k = \frac{1}{m!} \prod_{i \in \mathcal{M}} b_i \prod_{i \notin \mathcal{M}} (1-b_i) \prod_{j \in \mathcal{M}} b_j' \prod_{j \notin \mathcal{M}} (1-b_j') . \tag{8.1-3}$$

where $\mathcal{M}$ is the set of objects that are matched in this combination (containing m objects from each scene).

Now the ρ_k quantities will be considered. They can be obtained from the discrepancies between object parameters for a particular matching, provided that the probability distribution of the measurements of these parameters is known. It is assumed here that these have the Gaussian (normal) distribution.

In order to make the problem less nonlinear, instead of using θ and f as the parameters in the adjustment, the quantities c and s, defined as follows, are used:

$$\begin{aligned} c &= f \cos \theta \\ s &= f \sin \theta \\ \theta &= \arctan \frac{s}{c} \\ f &= \sqrt{c^2 + s^2} \end{aligned} \tag{8.1-4}$$

Now the transformation between object parameters in the two scenes for the special case under consideration can be expressed as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$\begin{bmatrix} w_{xx}' & w_{xy}' & w_{xz}' \\ w_{xy}' & w_{yy}' & w_{yz}' \\ w_{xz}' & w_{yz}' & w_{zz}' \end{bmatrix} = \frac{1}{f^4} \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & f \end{bmatrix} \begin{bmatrix} w_{xx} & w_{xy} & w_{xz} \\ w_{xy} & w_{yy} & w_{yz} \\ w_{xz} & w_{yz} & w_{zz} \end{bmatrix} \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & f \end{bmatrix}^{\mathrm{T}} \tag{8.1-5}$$

From this a rotation and scale factor matrix R can be derived to be

$$R = \frac{1}{f^4}\begin{bmatrix} f^4c & f^4s & 0 & 0 & 0 & 0 & 0 & 0 \\ -f^4s & f^4c & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c^2 & s^2 & 0 & 2cs & 0 & 0 \\ 0 & 0 & s^2 & c^2 & 0 & -2cs & 0 & 0 \\ 0 & 0 & 0 & 0 & f^2 & 0 & 0 & 0 \\ 0 & 0 & -cs & cs & 0 & c^2-s^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & fc & fs \\ 0 & 0 & 0 & 0 & 0 & 0 & -fs & fc \end{bmatrix} \qquad (8.1\text{-}6)$$

where it is assumed that $X = [x\ y\ w_{xx}\ w_{yy}\ w_{zz}\ w_{xy}\ w_{xz}\ w_{yz}]^{\mathrm{T}}$. (The horizontal components of **c** are denoted here by x and y, and the elements of W are denoted by w with appropriate subscripts.) In other cases using different features than the object descriptions used here, R would be defined differently. But in any case, it would be used as follows to compute a discrepancy vector Ξ and its covariance matrix Ψ for each matching of object i in Scene 1 with object j in Scene 2:

$$\begin{aligned} \Xi_{ij} &= RX_i - X'_j \\ \Psi_{ij} &= RS_iR^{\mathrm{T}} + S'_j \end{aligned} \qquad (8.1\text{-}7)$$

Now the partial derivatives of Ξ_{ij} with respect to the parameters c, s, $-x'$, and $-y'$ can be obtained from the above equation (by using the fact that $f^2 = c^2 + s^2$) and are assembled into the matrix B_{ij} (8 by 4 here). (The derivatives relative to $-x'$ and $-y'$ produce the effects relative to Δx and Δy, according to (8.1-5).) Then, using the general solution in Appendix A produces

$$\begin{aligned} H_k &= B_o^{\mathrm{T}}\begin{bmatrix} \sigma^2_{\theta_o} & 0 \\ 0 & \sigma^2_{f_o} \end{bmatrix}^{-1} B_o + \sum_{i,j\in\mathfrak{K}} B_{ij}^{\mathrm{T}}\Psi_{ij}^{-1}B_{ij} \\ C_k &= B_o^{\mathrm{T}}\begin{bmatrix} \sigma^2_{\theta_o} & 0 \\ 0 & \sigma^2_{f_o} \end{bmatrix}^{-1} \begin{bmatrix} \theta_o - \theta \\ f_o - f \end{bmatrix} + \sum_{i,j\in\mathfrak{K}} B_{ij}^{\mathrm{T}}\Psi_{ij}^{-1}\Xi_{ij} \end{aligned} \qquad (8.1\text{-}8)$$

where $\mathfrak{K}$ is the set of objects making up the kth combination, the first terms contain the a priori information, and

$$B_0 = \begin{bmatrix} -\frac{s}{f^2} & \frac{c}{f^2} & 0 & 0 \\ \frac{c}{f} & \frac{s}{f} & 0 & 0 \end{bmatrix} \tag{8.1-9}$$

which consists of the partial derivatives of θ and f with respect to c, s, Δx, and Δy, according to (8.1-4). Then the solution is obtained by replacing the values of c, s, Δx, and Δy by values obtained from the old values, as follows:

$$\begin{bmatrix} c \\ s \\ \Delta x \\ \Delta y \end{bmatrix} \leftarrow \begin{bmatrix} c \\ s \\ 0 \\ 0 \end{bmatrix} + H_k^{-1} C_k \tag{8.1-10}$$

H_k^{-1} is the covariance matrix of $[c\ s\ \Delta x\ \Delta y]^T$. Because of the nonlinear terms in (8.1-6) with respect to c and s, this process may need to be iterated. (Since the problem is linear with respect to Δx and Δy, these quantities did not have to be included in (8.1-7), and thus in (8.1-10) their old values are in effect zero.) Note that the nonlinearity with respect to c and s occurs in R only in the terms involving the w's. When the objects are considerably smaller than their separation or are nearly spherical, these terms have very little effect in the solution in (8.1-10). In such a case, if c and s are fairly accurately known, it would be a reasonable approximation to neglect their variation in B_{ij}, and thus only the terms involving B_0 in (8.1-8) (containing the a priori information) would have to be recomputed in the iterations.

Now the probability density for this particular matching combination can be obtained as follows for use in (8.1-2). First, the residuals are

$$V_{ij} = \Xi_{ij} - B_{ij} H_k^{-1} C_k \tag{8.1-11}$$

Then the multivariate normal distribution produces

$$\rho_{ij} = \frac{1}{(2\pi)^{r/2} \sqrt{\det(\Psi_{ij})}} \exp\left(-\frac{1}{2} V_{ij}^T \Psi_{ij}^{-1} V_{ij}\right) \tag{8.1-12}$$

where r is the number of parameters in the feature vector (size of V), here assumed to be 8. If object i is unmatched in this combination, the a priori distribution of parameter values is used for ρ_{ij} instead of (8.1-12). Finally, the probability density function for the complete match is the product of all of the ρ_{ij} values in this combination times the probability density functions corresponding to the a priori values, as follows:

$$\rho_k = \frac{1}{2\pi\sigma_{\theta_o}\sigma_{f_o}} \exp\left(-\frac{(\theta-\theta_o)^2}{2\sigma^2_{\theta_o}} - \frac{(f-f_o)^2}{2\sigma^2_{f_o}}\right) \prod_{i,j\in R} \rho_{ij} \qquad (8.1\text{-}13)$$

8.2 Search Procedure

With a large number of objects, it would be impractical to use all combinations in (8.1-2). However, because of the exponential function in (8.1-12), most of the ρ_k values will be negligibly small, and these terms can be ignored. The problem is to determine which combinations will produce significant magnitude in ρ_k, without having to compute them all.

The approach used is to select the objects in Scene 1 one at a time and to tentatively match these to all objects in Scene 2. The a posteriori probability of each of these partial combinations is computed, and those with negligibly small probability are not pursued further. In order to have a high likelihood of unambiguous matching, the larger objects in Scene 1 are selected first, although a more complicated ordering using the covariance matrices also could be devised. This successive matching of features in order to refine a transformation between scenes is similar in some respects to other scene-matching methods, for example Price [1978] and Milgram and Bjorklund [1979]. However, these did not use a full search, matching features one at a time, and did not use probabilities to prune the search.

Because, when a tentative partial match is made, it is not known which objects in Scene 2 will remain unmatched, there is no obvious way to use all of the information in (8.1-3). Instead, π_k is computed by the following method for the purposes of the search. (After the process has used all of the objects in Scene 1, the remaining complete combinations with significant probability can be used in the full computation described in the previous section.) The value of π_k is obtained recursively, as follows:

$$\pi_o = 1$$

$$\beta_o = \sum_j b'_j$$

$$\pi_k = \pi_{k-1} b_i \frac{b'_j}{\beta_{k-1}}, \quad \text{if } i \text{ and } j \text{ are matched} \qquad (8.2\text{-}1)$$

$$\pi_k = \pi_{k-1}(1-b_i), \quad \text{if } i \text{ is not matched}$$

$$\beta_k = \beta_{k-1} - b'_j, \quad \text{if } i \text{ and } j \text{ are matched}$$

$$\beta_k = \beta_{k-1}, \quad \text{if } i \text{ is not matched}$$

where k denotes the number of objects in Scene 1 that have been selected for matching so far. In this way a search tree is built up, branching out as different objects in Scene 2 (including no object) are matched with the current object in Scene 1 at each level. The values from (8.2-1) can be used in (8.1-2) as before, and the resulting probabilities are used to prune the search tree.

The criterion used for pruning ideally should not use a constant probability threshold, but should take into account the fact that, if a large number of nodes have small probability each but sum to a large probability, there is a good chance that one of them will turn out to be part of the correct solution. An appropriate method would be to sort by probability all of the nodes at a given level in the search and to reject all of the ones with smallest probabilities that sum to less than some threshold. The implemented version is simpler than this (and more tolerant); it rejects any node whose probability times the number of nodes at this level is less than the threshold. The threshold currently used is 0.001. This also is quite tolerant. However, once about two object matches are included in a combination, new matches usually do not agree very well unless they are correct, and thus the probability drops rapidly for incorrect combinations.

As the bottom level of the search tree is reached, not all of the available information will have been used in computing π_k by the above method. Thus π_k from (8.2-1) will act as an upper limit to the value that would have been obtained from (8.1-3). This effect causes less pruning to occur than the optimum computation would produce, but it should not result in the rejection of good solutions.

At each level of the search, the complete computation according to (8.1-8) and (8.1-10) can be computed, including the iterations. However, the result from the previous level in the search tree can be used as the initial approximation at this level, so that fewer iterations (perhaps only one) would be needed at each level. Also, as pointed out in section 8.1, under some conditions the variation in the elements of B_{ij} with respect to c and s can be ignored. In such a case the summations involving B_{ij} in (8.1-8) can be computed recursively by adding the terms for a new i,j combination to the total accumulated at a higher level in the search, thus saving time. If the a priori values of c and s were known so accurately that the variation in B_0 could also be ignored, then the entire computation of H_k and C_k could be done recursively, and it would be possible to reformulate the solution by using a mathematically equivalent Kalman [1960] recursive estimation technique (called sequential adjustment by Mikhail [1976]), with slight additional time savings.

It would also be desirable to save time by recursively computing the product of ρ_{ij} used to obtain ρ_k in (8.1-13). However, as formulated in section 8.1, the residuals V_{ij} vary with the current solution as more combinations are added. It is possible to reformulate things so that when a new combination is introduced, it is compared to the previously accumulated solution for the purposes of computing ρ_{ij}. When ρ_{ij} is defined in this way, its values do not change as more combinations are added, but its product with a given set of combinations is the same as with the other definition. Thus, its product can be accumulated recursively. However, this method would involve including the current estimate of translation in (8.1-7) and doing the entire error propagation from the current estimates of c, s, Δx, and Δy in (8.1-7) for these purposes. This would be rather complicated and time consuming, so it is not considered further here. However, it is described in the next section for an approximate method.

In any case, the solution at the bottom level can be used as an initial approximation for one or more iterations with the full computation described in the previous section, using those combinations that have not been pruned because of small probability. (However, this is not done in the implemented version.)

8.3 Approximations

The computations previously described, especially (8.1-8), would be quite time-consuming. Therefore, it is desirable to make some time-saving approximations, especially in the search phase where the computations will be repeated many times. The approximations that will be considered are mainly those that discard some of the information about the objects. The fact that such a process does not use all of the available information about the objects will result in less effective pruning. Thus, although less computation has to be done at each node in the search tree, there are more nodes to compute. (This is similar to the effect of using the approximate value of n_k in the previous section.) If desired, the full computation can be done with the nodes remaining at the bottom level, so that the approximations affect only the computation time and not the final result.

The approximation that is made in the search portion of the implemented version is to use only the position and size of each object and to disregard its shape and orientation. Also, the size is used only in computing the probability density for a given fit, and not in adjusting the parameters of the fit. (This latter change usually has little effect, because the scale factor is determined mainly by the distance between objects.) The quantity used to represent size is the trace of the W matrix, denoted here by t. (Actually t is inversely proportional to the square of a linear dimension of the ellipsoid.) The trace is chosen to represent size because it is easy to compute from the W matrix and it is invariant under rotations. Thus,

$$\begin{aligned} t &= w_{xx} + w_{yy} + w_{zz} \\ \sigma_t^2 &= \sigma^2_{w_{xx}} + \sigma^2_{w_{yy}} + \sigma^2_{w_{zz}} + 2\sigma_{w_{xx}w_{yy}} + 2\sigma_{w_{xx}w_{zz}} + 2\sigma_{w_{yy}w_{zz}} \end{aligned} \qquad (8.3\text{-}1)$$

Because none of the orientation information about the objects is being used, the only portion of the X vector in (8.1-7) that will contribute information to the solution in (8.1-8) is the position information. If only the horizontal position is used, then only the first two components of X need to be used in (8.1-7).

Thus the nonlinear terms in c and s have been eliminated from R, and B_{ij} is now independent of these parameters. Therefore, the recursive approach suggested in section 8.2 can be used in order to save time in the search phase. This involves updating the old probabilities when a new match is introduced into the combination. Thus the probability density of the new match is computed by comparing the new match to the previously computed solution, instead of examining the residuals of the overall solution as in (8.1-12). Therefore, for the purposes of computing the probability density only, the discrepancies are redefined to included the current estimates of translation (Δx and Δy), and the accuracy estimates of the discrepancies include the current uncertainty in the translation and rotation.

Making the above changes to Ξ in (8.1-7) and multiplying out the matrices produces

$$\begin{aligned} \xi &= xc + ys + \Delta x - x' \\ \upsilon &= yc - xs + \Delta y - y' \end{aligned} \qquad (8.3\text{-}2)$$

The full error propagation associated with this produces

$$\begin{aligned} \sigma_\xi^2 &= \sigma_x^2 c^2 + x^2\sigma_c^2 + 2\sigma_{xy}cs + 2xy\sigma_{cs} + \sigma_y^2 s^2 + y^2\sigma_s^2 + 2x\sigma_{c\Delta x} + 2y\sigma_{s\Delta x} + \sigma_{\Delta x}^2 + \sigma_{x'}^2 \\ \sigma_\upsilon^2 &= \sigma_y^2 c^2 + y^2\sigma_c^2 - 2\sigma_{xy}cs - 2xy\sigma_{cs} + \sigma_x^2 s^2 + x^2\sigma_s^2 + 2y\sigma_{c\Delta y} - 2x\sigma_{s\Delta y} + \sigma_{\Delta y}^2 + \sigma_{y'}^2 \\ \sigma_{\xi\upsilon} &= \sigma_{xy}c^2 + xy\sigma_c^2 - \sigma_x^2 cs - x^2\sigma_{cs} + \sigma_y^2 cs + y^2\sigma_{cs} - \sigma_{xy}s^2 - xy\sigma_s^2 \\ &\quad + \sigma_{\Delta x\Delta y} + x\sigma_{c\Delta y} + y\sigma_{c\Delta x} + y\sigma_{s\Delta y} - x\sigma_{s\Delta x} + \sigma_{x'y'} \end{aligned} \qquad (8.3\text{-}3)$$

The corresponding quantities for size are

$$\begin{aligned} \tau &= \frac{t}{f^2} - t' \\ \sigma_\tau^2 &= \frac{\sigma_t^2}{f^4} + \frac{4\sigma_f^2}{f^6}t^2 + \sigma_{t'}^2 \end{aligned} \qquad (8.3\text{-}4)$$

Then the probability density is computed as follows, derived from (8.1-12):

$$\rho_{ij} = \frac{1}{(2\pi)^{3/2}\sigma_\tau\sqrt{\sigma_\xi^2\sigma_\upsilon^2 - \sigma_{\xi\upsilon}^2}} \exp\left(-\frac{1}{2}\cdot\frac{\xi^2\sigma_\upsilon^2 - 2\xi\upsilon\sigma_{\xi\upsilon} + \upsilon^2\sigma_\xi^2}{\sigma_\xi^2\sigma_\upsilon^2 - \sigma_{\xi\upsilon}^2} - \frac{\tau^2}{2\sigma_\tau^2}\right) \qquad (8.3\text{-}5)$$

However, if no objects have been matched at previous levels, no values of Δx and Δy would have been computed for the above computation, so only the size information would be used, as follows:

$$\rho_{ij} = \frac{1}{af^2\sqrt{2\pi}\,\sigma_\tau} \exp\left(-\frac{\tau^2}{2\sigma_\tau^2}\right) \tag{8.3-6}$$

where a is the a priori area over which the objects might lie in Scene 1. In any case, the overall probabiltiy density for the match so far accumulated is the product of these:

$$\rho_k = \prod_{i,j\in\Re} \rho_{ij} \tag{8.3-7}$$

These values are used in (8.1-2), as before. In the search phase, if the resulting probability is small, the combination just produced is deleted. Otherwise, a new adjustment for the parameters is done.

Since the portion of (8.1-7) dealing with W is not used in the parameter adjustment in this approximation, the adjustment is linear except for the a priori values. The B matrix is

$$B = \begin{bmatrix} x & y & 1 & 0 \\ y & -x & 0 & 1 \end{bmatrix} \tag{8.3-8}$$

The error propagation from the position values according to (8.1-7) is as follows, which differs from (8.3-3) in that it does not include the uncertainty in the parameters being adjusted (c, s, Δx, and Δy):

$$\begin{aligned} \sigma_\xi^2 &= \sigma_x^2 c^2 + 2\sigma_{xy} cs + \sigma_y^2 s^2 + \sigma_{x'}^2, \\ \sigma_\upsilon^2 &= \sigma_y^2 c^2 - 2\sigma_{xy} cs + \sigma_x^2 s^2 + \sigma_{y'}^2, \\ \sigma_{\xi\upsilon} &= \sigma_{xy} c^2 - \sigma_x^2 cs + \sigma_y^2 cs - \sigma_{xy} s^2 + \sigma_{x'y'} \end{aligned} \tag{8.3-9}$$

Then the covariance matrix of the observations is

$$\Psi = \begin{bmatrix} \sigma_\xi^2 & \sigma_{\xi\upsilon} \\ \sigma_{\xi\upsilon} & \sigma_\upsilon^2 \end{bmatrix} \tag{8.3-10}$$

B and Ψ are used in (8.1-8). Note that B now is not a function of the parameters being adjusted (c, s, Δx, and Δy); however, Ψ is a function of c and s. The effect of Ψ changing slightly is only to change the weights of the observations slightly. Therefore, as long as c

and s do not change very much, there should be no need to recompute the summations involving these matrices when iterating. In the search phase, these summations can be accumulated recursively as more matches are added to the combination. The implemented program currently does it this way. Therefore, the a priori values θ_0 and f_0 must be fairly accurate, so that c and s will not change very much. (The implemented program also uses these a priori values in a less optimum way than is described in Section 8.1, and this has the same result of requiring the a priori values to be fairly accurate.)

Other approximations are possible that use more information than the above but less than the full information available. One possibility is to use the position, size, and shape of each object but to ignore its orientation. (If the a priori orientation is completely unknown, when only one object match exists in a combination this is all the information that is useful anyway, but as more objects are matched in a search, the orientation information may become useful.) This approximation would be used in a similar manner to the above approximation, except that there would be three size and shape quantities instead of t, with a 3-by-3 covariance matrix instead of σ_t^2. Like t, these quantities affect the probability of a match, but their effect on the adjustment for c, s, Δx, and Δy for a particular match is small and can be ignored.

The size and shape of an ellipsoid are determined by the semi-lengths of its principal axes. These are equal to the reciprocal of the square roots of the eigenvalues of the W matrix. The eigenvalues of the W matrix can be found by solving the following equation for λ (see Hohn [1973]):

$$\det(W - \lambda I) = 0 \qquad 8.3\text{-}11)$$

where I is the 3-by-3 identity matrix. Since W is 3-by-3, (8.3-11) is a cubic in λ, and the three roots are the three eigenvalues. The most difficult part is not computing the eigenvalues themselves, but computing the error propagation from the elements of W to the eigenvalues. For a linear approximation error propagation it is necessary to compute (either analyticaly or numerically) the partial derivarives of the eigenvalues with respect to the six unique elements of W. Then the error propagation is done in the usual way by forming these partial derivatives into a 3-by-6 matrix, premultiplying the 6-by-6 covariance matrix of the w's by this matrix, and postmultiplying by its transpose to produce the covariance matrix of the eigenvalues.

8.4 Example

The only pictures used to test the scene matcher were the Mars pictures described in Appendix C. Ideally, different stereo pairs taken under different lighting conditions and from different directions should have been used to obtain the scene descriptions to be matched, in order to have a more impressive test. However, the Viking could take pictures from only one position, and because of a lack of time pictures taken under different conditions were not transferred to our computer system. However, in order to simulate the differences that might occur from these causes, the pictures used were processed in different ways. The small portions (about 10° by 10°) were processed with 8-by-8 match windows in the stereo program and with height thresholds of 3, 5, 6, and 7 centimeters in the object finder; the small portions were also processed with 5-by-5 match windows and a 5-centimeter threshold; and the large portions were processed with 8-by-8 match windows and a 6-centimeter threshold. The results were compared to each other successfully by the scene matcher. One of these matches is shown here.

Figure 8-1 and Figure 8-2 repeat Figures 7-7 and 7-8, except that the objects have been identified with arbitrary numbers. (These scene descriptions were obtained with height thresholds of 5 centimeters and 3 centimeters, respectively, in the object finder. The match window in the stereo processing was 8 pixels wide in both cases.) These two scene descriptions were given to the scene matcher. The a priori rotation was given as zero with a standard deviation of 1°, and the a priori scale factor was given as unity with a standard deviation of 0.01. (The translation was completely free to be adjusted.)

Figure 8-3 shows the results of using the data in Figure 8-1 as Scene 1 and the data in Figure 8-2 as Scene 2. The search tree is shown. The numbers at the left followed by colons are the object numbers in Scene 1. The other numbers on the same line are the object numbers in Scene 2 for the objects being matched to this object in Scene 1. Zero means that this object in Scene 1 is left unmatched. The numbers just below the Scene 2 object numbers represent the probabilities computed for this match so far. To save space the negative of the common logarithm of the probability, truncated to an integer, is shown. Below the search tree the final results are shown for the most probable match. Shown are the pairings of objects, the probability, the translation in x and y, the rotation, and the scale factor. The values after the plus-or-minus signs are the computed standard deviations. Note that the standard deviation of the scale factor is not much less than the input value of 0.01, which means that the solution was not able to add much information about the scale factor. (Since the two scenes were both from the same actual scene and same camera position, the true values of translation and rotation are zero, and the true value of scale factor is unity.)

Figure 8-4 similarly shows the results of doing the match with the scenes interchanged, so that the data in Figure 8-2 is now Scene 1 and the data in Figure 8-1 is Scene 2. Even though the search tree is quite different, the final results are almost the same. (The final results would be exactly the same if a complete solution as described in Section 8.1 were done at the bottom level of the search tree, since the same objects were matched.) Of course, since the scenes have been interchanged, the translation and rotation have changed sign and the scale factor has been inverted.

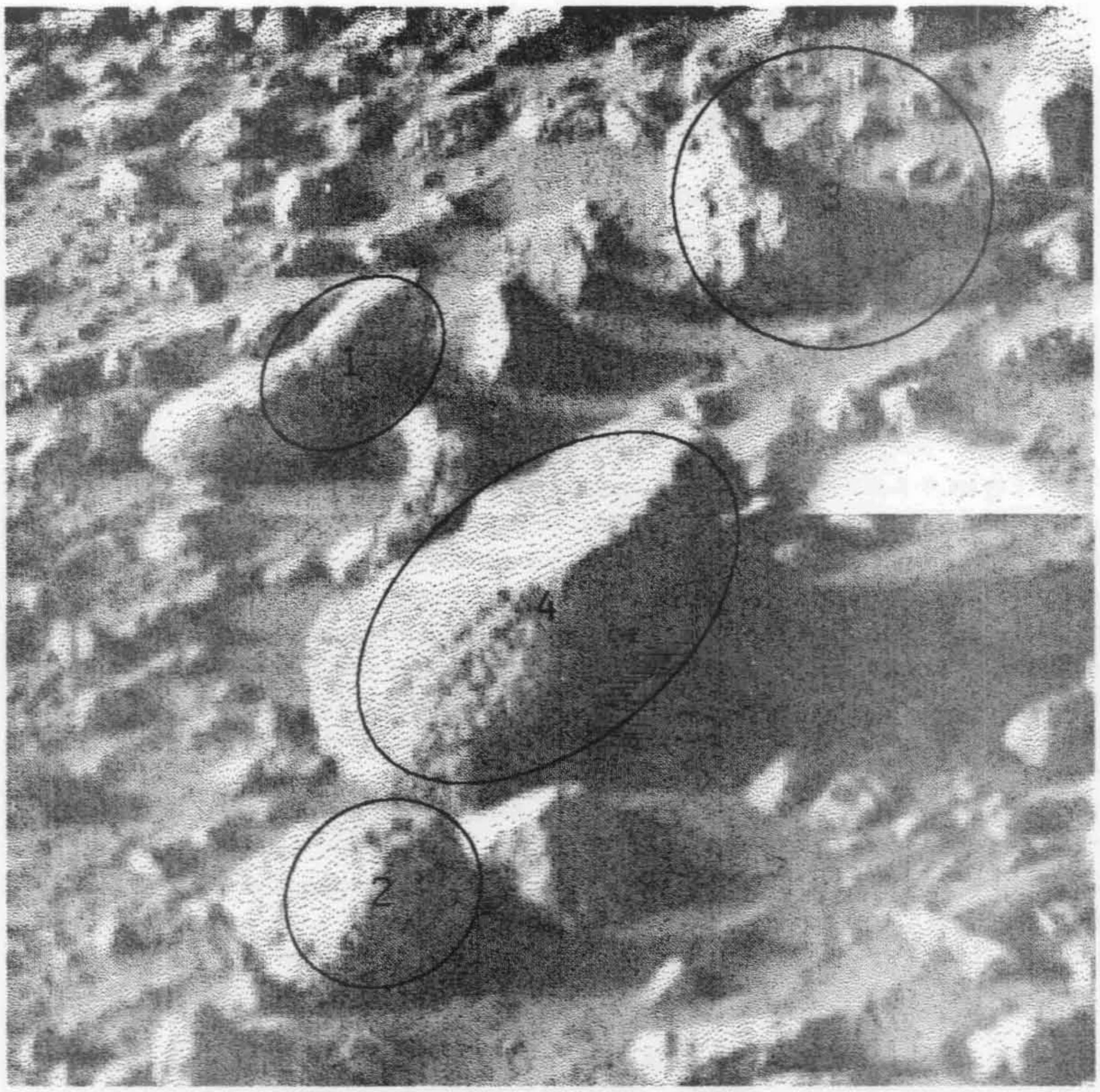

Figure 8-1. Ellipsoids produced with 5-cm height threshold.

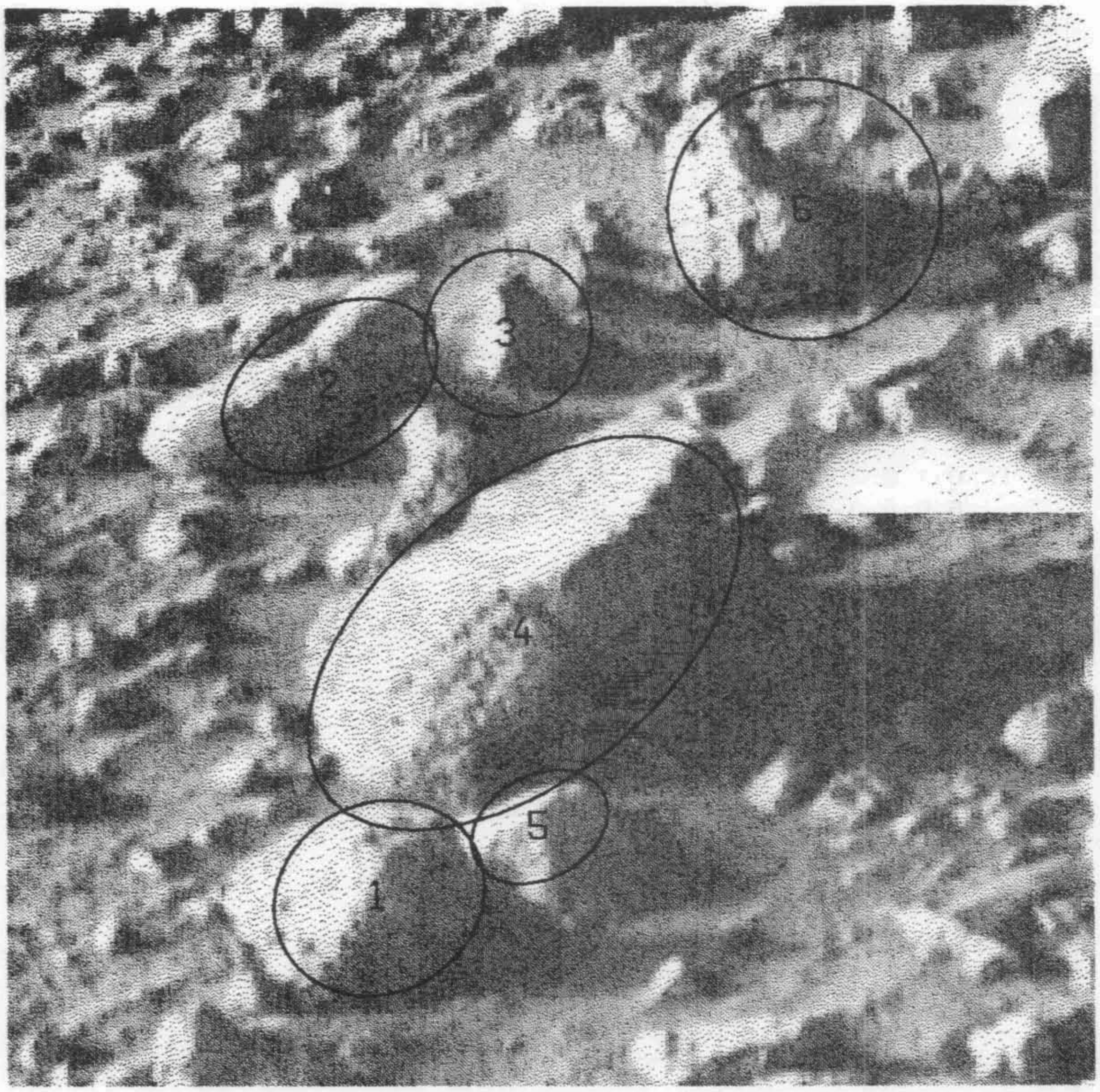

Figure 8-2. Ellipsoids produced with 3-cm height threshold.

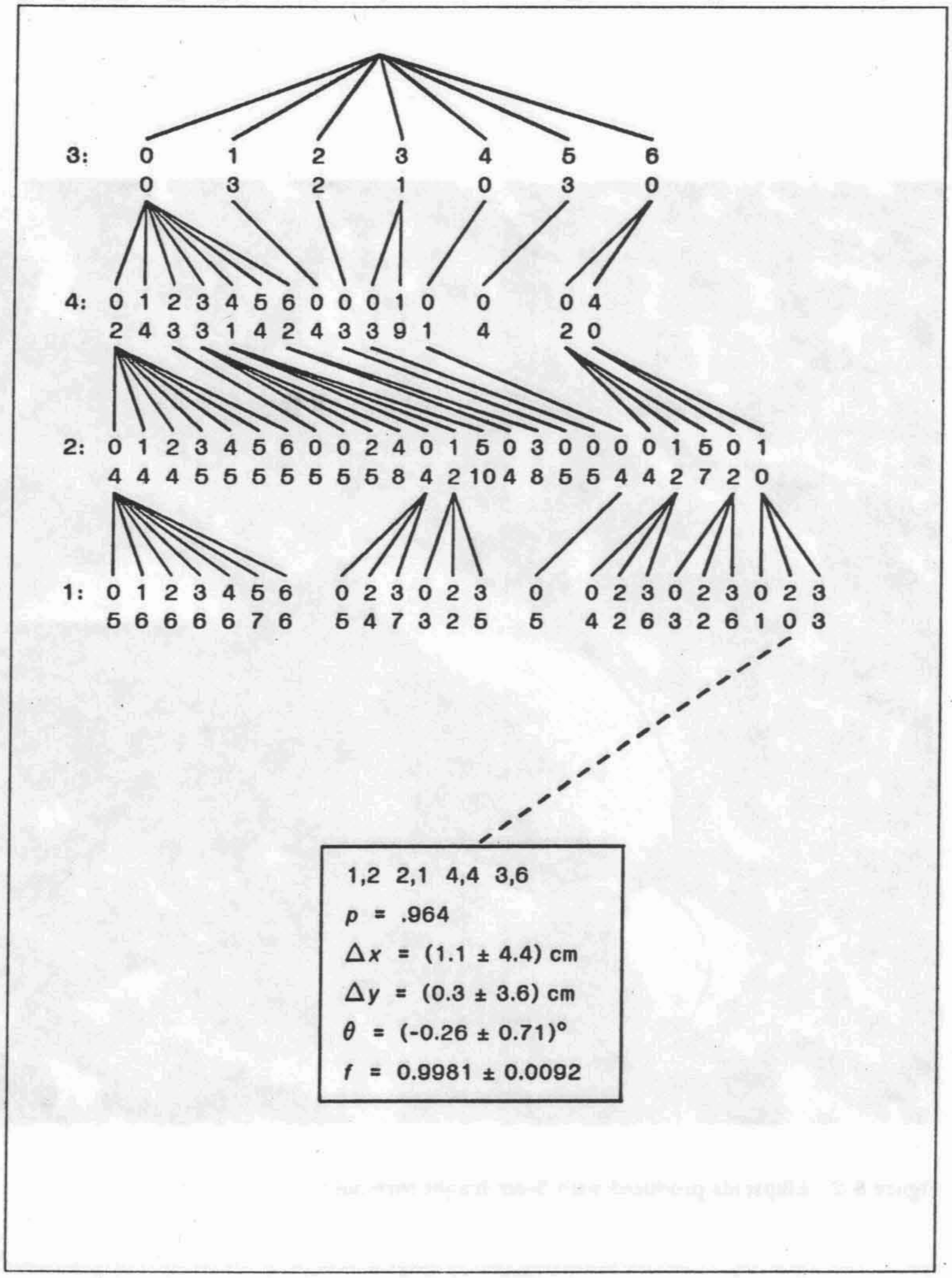

Figure 8-3. Match of scene in Figure 8-1 to scene in Figure 8-2.

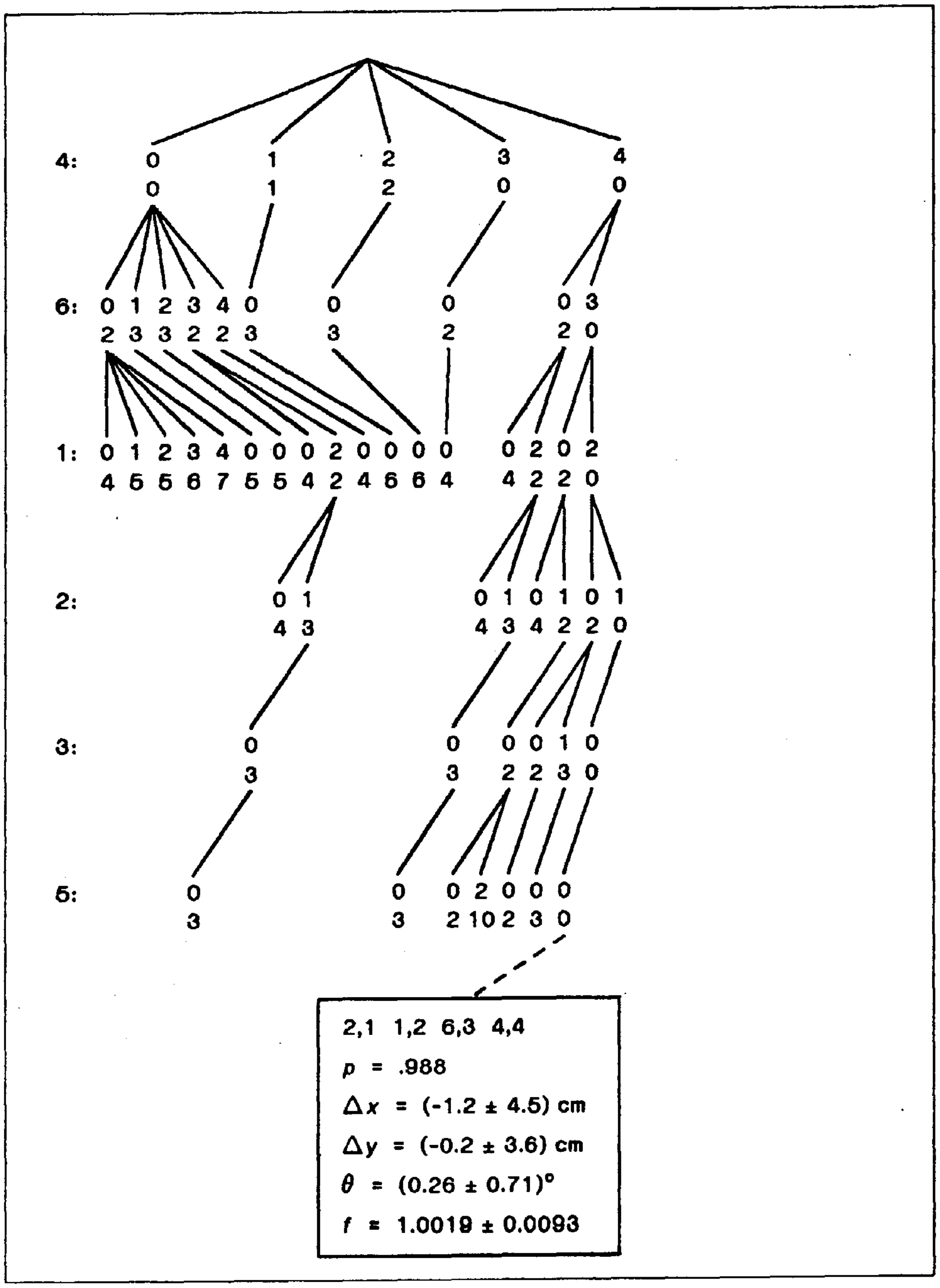

Figure 8-4. Match of scene in Figure 8-2 to scene in Figure 8-1.

Appendix A

NONLINEAR PARAMETER ADJUSTMENTS

In several parts of this thesis, problems have been discussed that involved solving for some parameters by using observed values of some quantities that are nonlinear functions of these parameters. The observed values may contain occasional mistakes (wild points), and the accuracy of the remaining values may not be entirely known. For example, in the stereo camera model solution, measured values of film plane coordinates of some points are obtained, and it is desired to compute the relative position and orientation of the cameras. A general method for solving problems of this type will be discussed in this appendix.

A.1 Basic Method

In this section a method of performing nonlinear generalized least-squares adjustments will be described. This method uses partial derivatives to linearize the problem and iterates to achieve the exact solution. It is assumed in this section that the accuracy of the observations is known and that there are no wild points that should be removed from the solution.

Very little in this section is original. However, this method apparently is not well known in Artificial Intelligence circles. Even in circles where the basic method is often used, some of its properties are not well known. For example, the linearized solution represented by (A.1-17) is often described (as in Mikhail [1976]) without any apparent awareness of the effects of the second derivatives. Bard [1974] mentions the second derivatives but does not cover the related matters dealt with here in the following sections. For these reasons it is desirable to describe here in a unified manner the method as it is used in this thesis. Of course, the two references just cited also deal with other aspects of this problem and similar problems not needed here. Also, more information about the statistical properties of the linear problem is given by Graybill [1961]. Therefore, the reader who is interested in these matters is urged to consult these references for further information.

Suppose we have a set of m unknown parameters for which values are desired, denoted by the vector G (m-by-1 matrix). (In the stereo camera model solution, these would be the quantities defining the camera calibration.) Suppose further that there are a set of n quantities ($n \geq m$) denoted by the vector F, which can be measured with some error and which are functions of G. Let U denote the measured value of F (containing some error). (In the stereo camera model solution, the elements of U would be related to the film plane measurements in a way explained in Section 3.3.) Let V be the vector of the n residual errors in the fit to the observations using a particular set of values for the parameters.

That is,

$$U = F(G) + V \tag{A.1-1}$$

with the functional dependence on G explicitly indicated. The problem is to use U to compute G such that V is minimized in some sense. (The n scalar equations represented by (A.1-1) are called the condition equations.)

For the criterion of optimization we will minimize the quadratic form

$$q = V^TWV \tag{A.1-2}$$

where W denotes an n-by-n weight matrix. W should be the inverse of the covariance matrix of the errors in the observations. This will result in the maximum-likelihood (in the F space) solution if the errors have the multivariate Gaussian (normal) distribution, and it will result in the minimum-variance (of the g's) unbiased solution of all solutions that are linear functions of the U, for any distribution in linear problems. (Proofs of these statements can be found in Graybill [1961].) Note that if W is a diagonal matrix (indicating no correlation between errors in different observations) the quadratic form reduces to a weighted sum of the squares of the elements of V. Thus the problem as stated here can be said to be a generalized least-squares adjustment.

Solving (A.1-1) for V and substituting in (A.1-2) produces

$$q = [U - F(G)]^TW[U - F(G)] \tag{A.1-3}$$

The problem then is to find G such that q is minimum.

The difficulty in obtaining a solution to the above problem lies in the fact that F in (A.1-1) is a nonlinear function, and thus in general there is no closed form solution. One way of solving the problem is to use some type of general numerical minimization technique, in which on various iterations new values of G are tried, q is recomputed each time, and q is driven to a minimum. However, such methods tend to converge rather slowly. Also, numerical problems may occur if q has a very broad minimum, for round-off errors may give rise to spurious local minima. Instead of such an approach, to find the minimum of q, we will differentiate (A.1-3) with respect to G, set the result to zero, and solve for G iteratively. (A numerical value of q then never needs to be computed to obtain G.)

In order to follow the steps of this process, we rewrite (A.1-3) in terms of the elements of the matrices, as follows (where a particular element of a matrix is represented by the corresponding lower-case letter with appropriate integer subscripts):

$$q = \sum_{i,j} [u_i - f_i(G)]w_{ij}[u_j - f_j(G)] \tag{A.1-4}$$

Differentiating this produces

$$\frac{\partial q}{\partial g_k} = -2\sum_{i,j} \frac{\partial f_i}{\partial g_k} w_{ij}[u_j - f_j(G)] \tag{A.1-5}$$

Since (A.1-5) is a nonlinear equation, to solve it for G when $\partial q/\partial g_k$ is set to zero, we will use Newton's method. To do this, the partial derivatives of $\partial q/\partial g_k$ are needed. These are

$$\frac{\partial^2 q}{\partial g_k \partial g_l} = 2\sum_{i,j} \frac{\partial f_i}{\partial g_k} w_{ij} \frac{\partial f_j}{\partial g_l} - 2\sum_{i,j} \frac{\partial^2 f_i}{\partial g_k \partial g_l} w_{ij}[u_j - f_j] \tag{A.1-6}$$

The corrections d_i needed to g_i are related to the above by

$$\sum_j \frac{\partial^2 q}{\partial g_i \partial g_j} d_j = -\frac{\partial q}{\partial g_i} \tag{A.1-7}$$

(These corrections would be exactly correct if F were linear.) We can now revert to matrix notation, by defining the n-by-m matrix P to be composed of the partial derivatives of the function F, such that

$$p_{ij} = \frac{\partial f_i}{\partial g_j} \tag{A.1-8}$$

and the n-by-m-by-m matrix R to be composed of the second derivatives of F, such that

$$r_{ijk} = \frac{\partial^2 \partial f_i}{\partial g_j \partial g_k} \tag{A.1-9}$$

Substituting (A.1-5) and (A.1-6) into (A.1-7), using these definitions, and dividing through by 2 produces

$$[P^T W P - R^T W(U - F)]D = P^T W(U - F) \tag{A.1-10}$$

where F, P, and R are all implicit functions of G. (An approximate value of G used to obtain F, P, and R in (A.1-10) defines the correction D needed to obtain a more accurate value.) Notice that R is a strange creature, a three-dimensional matrix. These are not usually defined in matrix algebra, but the usual definitions can be generalized to handle them. In particular, a product of the form $A = R^T WB$, where A, R, W, and B have respectively two, three, two, and one dimensions, is given by $a_{kl} = \Sigma\, r_{ilk} w_{ij} b_j$, where the summation is over all values of i and j. (Of the five possible ways of rearranging the three indices, the transpose of a three-dimensional matrix is defined here as reversing the order of the three indices.)

The solution for D can expressed in terms of the matrix inverse as follows:

$$D = [P^{T}WP - R^{T}W(U - F)]^{-1}P^{T}W(U - F) \qquad (A.1\text{-}11)$$

or equivalently

$$D = [I - (P^{T}WP)^{-1}R^{T}W(U - F)]^{-1}(P^{T}WP)^{-1}P^{T}W(U - F) \qquad (A.1\text{-}12)$$

where I denotes the identity matrix (in this case m-by-m). D as obtained above using an approximate value of G would be added to this value of G to obtain a more accurate value, and this process would repeat until it converged.

The worst part of the above solution is the necessity to compute the partial derivatives. Often they are difficult to derive analytically and difficult to compute accurately numerically. In either case they are time-consuming to compute. These difficulties are usually much worse for the second derivatives R than for the first derivatives P. Furthermore, there are nm^2 second derivatives to compute and only nm first derivatives. Therefore, it is highly desirable to be able to omit the second derivatives from the computation. We will now consider the effect of neglecting them.

With a reasonable first approximation, and especially on later iterations, the discrepancies $U-F$ are small. Also, if the function F is reasonably smooth, the second derivatives R are small. Of course, what is considered small is relative. In this case smallness depends on the magnitude of the first derivatives P. If $U-F$ and R are small enough so that the relative change in P is small when G changes enough to cause F to vary by amounts on the order of $U-F$, then the nonlinearities are not having much effect, and the elements of $R^{T}W(U-F)$ are small compared to the elements of $P^{T}WP$. Thus a good approximation in such cases can be obtained by setting R to zero in (A.1-11) or (A.1-12), which produces

$$D \approx (P^{T}WP)^{-1}P^{T}W(U - F) \qquad (A.1\text{-}13)$$

The use of this approximation is known as the Gauss method, because Gauss originally used it on ordinary least-squares problems.

It is important to realize that only the second derivatives of F are neglected in the Gauss method. The second derivatives of q depend not only on these but also on the first derivatives according to (A.1-6). Under the assumptions in the previous paragraph the first term on the right of (A.1-6) usually is considerably larger than the second term, and thus the second derivatives of q will be fairly accurate.

The approximate (Gauss) corrections given by (A.1-13) are just the accurate (Newton) corrections given by (A.1-11) or (A.1-12) premultiplied by $I - (P^TWP)^{-1}R^TW(U-F)$. The accurate corrections given by (A.1-11) or (A.1-12) attempt to nullify an error in G which Newton's method has estimated to be $-D$, since $-D+D = 0$. But, if the Gauss method is used instead, we have in effect $-D + (I-A)D = -AD$, so that the vector of errors in G on each iteration is premultiplied by $A = (P^TWP)^{-1}R^TW(U-F)$, neglecting the higher order effects neglected in Newton's method. Therefore, using the approximation (A.1-13) cannot effect the final solution, unless it destroys the convergence. The matrix $(P^TWP)^{-1}R^TW(U-F)$ will tend to become constant as the solution convergences, as the discrepancies $U-F$ converge to the final value of the residuals V. Thus the Gauss method changes the quadratic convergence of Newton's method to linear convergence, if convergence is achieved. If all of the eigenvalues of $(P^TWP)^{-1}R^TW(U-F)$ have an absolute value less than one, convergence will be preserved, and the smaller the eigenvalues are, the faster convergence will be. (After several iterations, the error will tend to decrease by a factor equal to the absolute value of the largest eigenvalue.) From the arguments in the previous paragraph, the eigenvalues should be small, except when the initial approximation is very wrong (causing $U-F$ to be large) or when F is very nonlinear (causing R to be large). Thus, except in these cases, the solution should converge rapidly. (A way of converting the linear convergence of the Gauss method into quadratic convergence without computing R will be discussed in a later section.) Some of these matters are discussed further by Bard [1974].

The solution using (A.1-13) is usually obtained by a different approach (as in Brown [1955 and 1957] and Mikhail [1976]). This approach approximates (A.1-1) by a linearization based on the partial derivatives of F, solves the resulting linear problem, and iterates this process to obtain the solution to the nonlinear problem. Thus let G_0 denote an approximation to G. Then equation (A.1-1) can be approximated as follows:

$$U = F(G_0) + P(G_0)(G - G_0) + V \qquad \text{(A.1-14)}$$

where P is defined by (A.1-8) and its functional dependence on G has been explicitly indicated. We now define

$$\begin{aligned} E &= U - F(G_0) \\ D &= G - G_0 \end{aligned} \qquad \text{(A.1-15)}$$

Then (A.1-14) can be rewritten as

$$E = PD + V \qquad \text{(A.1-16)}$$

Thus we have replaced the nonlinear equation (A.1-1) by the linear equation (A.1-16), in which E represents the discrepancy between the observations and their computed values using the current approximations of the parameters, and D represents the corrections needed to the parameters. Therefore, we now wish to solve for D in (A.1-16) so as to minimize q in (A.1-2). This is a standard problem in linear statistical models. (See, for example, Graybill [1961].) The solution for D is

$$D = (P^T W P)^{-1} P^T W E \tag{A.1-17}$$

which is the same as (A.1-13).

The covariance matrix S_G of the errors in the converged values of the parameters G can be obtained from the covariance matrix S_U of the errors in the observations U by the usual linear approximation of premultiplying by the matrix of partial derivatives of the transformation and postmultiplying by the transpose of this matrix. In this case the transformation from U to G in the neighborhood of the converged values is given by approximately (A.1-13) or more accurately by (A.1-12). (Regardless of which method was used to produce the converged values of G, the answer is the same. Thus the use of (A.1-12) will produce a more accurate error propagation than (A.1-13), although (A.1-12) is still only an approximation in this regard if higher-order terms are considered.)

If the accurate transformation (A.1-12) is used, the matrix of partial derivatives will contain terms produced when (A.1-12) is differentiated relative to both occurrences of U in (A.1-12). However, when the derivatives are evaluated at the converged values, the effect of the first term drops out, since $P^T W(U-F)$ is then zero (because D is then zero). Thus we have

$$S_G = [I - (P^T W P)^{-1} R^T W(U-F)]^{-1} (P^T W P)^{-1} P^T W S_U W P (P^T W P)^{-1} [I - (P^T W P)^{-1} R^T W(U-F)]^{-1,T} \tag{A.1-18}$$

If $W = S_U^{-1}$, as it should for the optimum solution, this reduces to

$$S_G = [I - (P^T W P)^{-1} R^T W(U - F)]^{-1} (P^T W P)^{-1} [I - (P^T W P)^{-1} R^T W(U - F)]^{-1,T} \tag{A.1-19}$$

Using the approximation of neglecting the second derivatives, as in (A.1-13), reduces this to

$$S_G = (P^T W P)^{-1} \tag{A.1-20}$$

(Remember that (A.1-19) and (A.1-20) are correct only if W is the inverse of the covariance matrix of the observation errors.)

Note that even though (A.1-19) was derived using the linear approximation for covariance propagation, it contains the second derivatives of F. An even more accurate result could be obtained by considering second-order effects in the propagation, although this would require knowledge of moments of the error distribution of higher order than the second. This result would contain squares and cross products of the second derivatives, whereas they occur to the first power in (A.1-19). Therefore, if the second derivatives are small, (A.1-20) and (A.1-19) can be considered the first two members of an infinite sequence of better approximations, accurate to higher powers of the second derivatives. In most cases (A.1-20) is quite adequate, since the error estimates usually are not known very accurately anyway.

It often is desired to know the covariance matrix of the residuals. (It is useful to compare the magnitude of the residuals to the square roots of the diagonal elements of their covariance matrix, for editing purposes, as will be described in a Section A.5.) For the approximate (Gauss) case, this can be derived by first obtaining the equation for the residuals by solving (A.1-16) for V, substituting (A.1-17) for D, and factoring out E, to produce

$$V = [I - P(P^TWP)^{-1}P^TW]E \qquad (A.1\text{-}21)$$

Then, since the covariance matrix of E is the same as that of U, the covariance matrix of V can be obtained by premultiplying S_U by the coefficient of E (in brackets) in (A.1-21) and postmultiplying it by the transpose of this coefficient. If $W = S_U^{-1}$, the resulting expression simplifies to

$$S_V = S_U - P(P^TWP)^{-1}P^T \qquad (A.1\text{-}22)$$

Note that by using (A.1-20) the second term in this equation is seen to be the covariance matrix of the adjusted parameters propagated into the observations; thus it is the covariance matrix of the adjusted observations. Therefore, (A.1-22) says that the covariance matrix of the residuals is equal to the covariance matrix of the observations minus the covariance matrix of the adjusted observations. This may seem appropriate, because the residuals are the observations minus the adjusted observations. However, this should be considered a coincidence, because the covariance matrix of the difference or sum of two vectors is the sum of their covariance matrices, not the difference, if the vectors are uncorrelated with each other. Here, the particular way in which the observations and the adjusted observations are correlated produces the above result. Turning this around and expressing the observations as the sum of the adjusted observations and the residuals (and similarly for their covariance matrices) produces the somewhat surprising result that the residuals are uncorrelated with the adjusted observations. (Remember that these results holds only in the Gauss approximation and only if the weight matrix is the inverse of the covariance matrix of the observations.)

In many cases W can be partitioned into a diagonal matrix of matrices. Let each of these submatrices on the main diagonal of W be denoted by W_i. In the corresponding manner E and P are partitioned by rows into E_i and P_i. (What we have done is to group the observations into sets so that there is no correlation of errors between members of different sets.) Then (A.1-17) and (A.1-20) can be rewritten as

$$
\begin{aligned}
H &= \sum_i P_i^T W_i P_i \\
C &= \sum_i P_i^T W_i E_i \\
D &= H^{-1} C \\
S_G &= H^{-1}
\end{aligned}
\qquad \text{(A.1-23)}
$$

Note that, if the errors in all of the observations are uncorrelated, W_i and E_i are 1-by-1 matrices, which can be represented as the scalars w_i and e_i, and P_i is a 1 by m matrix. Furthermore, if all of the w_i are equal, they cancel out of the equation for D, and we have an unweighted solution (ordinary least-squares).

Several other quantities of interest can be derived from the solution. We will present these for the general partitioned Gauss case, with $W_i = S_{U_i}^{-1}$. The adjusted value of E_i is $P_i D$. The residuals are

$$V_i = E_i - P_i D \qquad \text{(A.1-24)}$$

The quadratic form is

$$q = \sum_i V_i^T W_i V_i \qquad \text{(A.1-25)}$$

The expected value of q is $n-m$. If the scale factor of the covariance matrix of observation errors is unknown, W can be adjusted by the ratio $(n-m)/q$ and S_G by the ratio $q/(n-m)$. Otherwise, q can be used as a test on the adjustment; for, if the observation errors have the Gaussian distribution, then q has the chi-square distribution with $n-m$ degrees of freedom. (Proofs of these properties of q can be found in Graybill [1961].) S_G represents the covariance matrix of errors in the adjusted parameters. The square roots of the diagonal elements of S_G are the standard deviations of the adjusted parameters. The correlation matrix of the parameters can be obtained from S_G by dividing the i,j element by the product of the standard deviations of the ith and jth parameters, for all i and j. Other results which follow directly from the results for the unpartitioned case are the covariance matrix of the adjusted observations $P_i S_G P_i^T$ and the covariance matrix of the residuals $S_{U_i} - P_i S_G P_i^T$. (Some of these matters are discussed further by Brown [1955 and 1957].)

The solution of the nonlinear problem can now be described as follows. An initial approximation is used to compute the discrepancies E_i and the partial derivatives P_i. Then D is computed from (A.1-23) and is added to the current approximation for G to obtain a better approximation. This process repeats until there is no further appreciable change in G. Then the final values from the last iteration can be used to obtain S_G, V_i,

q, and the other derived quantities described above. Of course, for convergence to the absolute minimum of q rather than convergence to some local minimum or divergence, it is necessary that the initial approximation be sufficiently close to the true solution. In most practical problems this is not critical; in fact, often there is only one minimum.

As previously discussed, the above solution for G, when converged, produces the true generalized least-squares adjustment regardless of the nonlinearity. However, the properties that the solution for G is minimum-variance and unbiased are only approximate in the nonlinear case. Also, as previously discussed, S_G as computed above is only approximately the covariance matrix of the errors in the final value of G in the nonlinear case. However, if the amount of nonlinearity over the range of the measurement errors is small, these results will be fairly accurate.

Often it is desired to have observations directly on the parameters. There are several possible reasons for this. There may be some a priori information about the parameters that one wants to combine into the solution. Also, it may be desired to give the initial approximations a very small amount of weight in the solution, so that in case one of the parameters would otherwise be indeterminate, it will be constrained sufficiently to prevent the H matrix from being singular and thus to allow a solution for the other parameters to be obtained. Finally, it may be desired to remove a parameter from the adjustment and to constrain it to a fixed value. This can be done by assigning a very large weight to the given value (although it would save computer time to delete this quantity from the parameters in the program instead). In any of these cases the desired effect can be achieved by creating an additional m-by-m P_i matrix, say P_0, equal to the identity matrix. Corresponding to this there is E_0, equal to the given a priori value of G minus the current approximation of G, and an m-by-m matrix W_0, the desired a priori weight matrix. These are included in the summations for H and C just like any other observations.

A few comments should be made about the numerical aspects of performing the computations. The H matrix is always non-negative definite; that is, if it is not singular it is positive definite. The best strategy to use when inverting a positive-definite matrix by an elimination technique is to pivot on the main diagonal. (See Forsythe and Moler [1967].) Therefore, a simple matrix inverter without any pivoting can be used to obtain H^{-1}. H is also symmetrical; therefore, some computation time can be saved if an inverter which makes use of this fact is used. However, if n is considerably larger than m, much more time is spent in computing H than in inverting it, so this may be hardly worth the trouble. In problems where the solution is nearly indeterminate, H will be nearly singular, and much accuracy can be lost because of numerical roundoff error. In such cases it may be necessary to use double precision in the computations for H, C, D, and S_G according to (A.1-23), including the inversion of H. (If a good inverter is used, there is usually not much point in having it in double precision unless a double-precision H is available to invert, as explained by Forsythe and Moler [1967].) However, high precision is not needed in computing the discrepancies E_i and the partial derivatives P_i, as long as consistent values are used throughout the computations for H and C.

A.2 Correlated Errors

The solution presented in the previous section allows the observation errors to be correlated in an arbitrary way, as represented by the covariance matrix S_U. However, in order to partition the solution correctly according to A.1-23, the errors in the different groups must be uncorrelated. But because of the great savings of time and storage that the partitioned solution allows, it sometimes is desirable to approximate the complete solution by means of the partitioned solution, even though the errors are correlated. This section describes a way in which this can be done under some circumstances.

It sometimes is the case that the observations are performed at points distributed throughout some space, with the covariance of different points always being less than the variance of any point and being negligible for points so far apart that their effects on the solution are significantly different (that is, have significantly different P_i matrices). For example, in the stereo camera model solution described in Chapter 3, the covariance between points is caused by the additional errors described in Section 3.2, whose covariance is a function of the distance between the points in the image plane and is assumed to be negligible for far-apart points.

In such a case the following approximation can be made. The covariance matrix S_U is partitioned into the covariance matrices S_{ij} of each pair of points i and j. (In the case of the camera model adjustment, each individual covariance matrix S_{ij} for points i and j is then 2-by-2.) Then an artificial covariance matrix for each point is computed as follows:

$$\hat{S}_{ii} = \sum_j S_{ij} \tag{A.2-1}$$

and all $\hat{S}_{ij}$ are assumed to be zero for $i \neq j$. The results from (A.2-1) are inverted to produce W_i for each point.

To see why this approximation works, consider the following extreme case, where the assumptions apply either to the entire covariance matrix S_U or to each of its submatrices partitioned (in the usual way for (A.1-23)) into groups of points with no correlation between groups. Assume that when the covariance matrix is partitioned further into submatrices corresponding to the points, all of the these submatrices on the diagonal are equal, all of the off-diagonal submatrices are equal, and all of the corresponding submatrices of P are equal (within a given group).

Under the above assumptions, the covariance matrix of each group can be considered to be made up of submatrices corresponding to the points such that the main-diagonal submatrices are $A + B$ and the off-diagonal submatrices are B. Then by multiplying the matrices it is easily verified that the inverse of the covariance matrix of the group of points contains $A^{-1} - (A+n_pB)^{-1}BA^{-1}$ everywhere on the diagonal and $-(A+n_pB)^{-1}BA^{-1}$ everywhere off the diagonal, where n_p is the number of points in the group.

Because of the assumption about P, the constant submatrices of P (denoted here by P_0) allow the terms for this group in the summations for H and C in (A.1-23) (or the corresponding portions of the unpartitioned solution in (A.1-17)) to be factored into $P_0^{\mathrm{T}}(\Sigma W_{ij})P_0$ and $P_0^{\mathrm{T}}(\Sigma W_{ij}E_j)$, respectively, where the summations are over all values of i and j. (The index i here should not be confused with i in (A.1-23), which is for the higher-level of partitioning, if any.) Thus it can be seen that the group of points is equivalent to one point which is the weighted average of the points, where the weight matrix for each group is the sum of the row (or column, since the matrix is symmetrical) of weight submatrices corresponding to this point. From the previous paragraph the sum of a row of submatrices of the inverse of the covariance matrix is seen to be $A^{-1} - n_p(A+n_pB)^{-1}BA^{-1}$, which simplifies to $(A+n_pB)^{-1}$. Under the approximation of (A.2-1) the artificial covariance matrix for this group consists of submatrices $A+n_pB$ on the diagonal and zero elsewhere. Inverting this and summing over the row produces $(A+n_pB)^{-1}$ (since there is only one term in the summation). This is equal to the exact result just produced. Therefore, in this special case the approximation is exact.

For another limiting case in which the approximation is exact, consider the points to be equally spaced in a Euclidean space of an arbitrary number of dimensions, with the covariance between a pair of points a function only of the coordinates of one point relative to the other. Thus, in the nomenclature of time series analysis, the errors are said to be stationary, and the covariances form the autocovariance function. Summing the autocovariance according to (A.2-1) over all of the space produces the zero-frequency value of the Fourier transform of the autocovariance function, which is the power spectrum of the errors. (See Blackman and Tukey [1958].) Therefore, what we have done is to use the value of the power spectrum at zero frequency. Grenander [1954] and Watson [1967] have shown that the component of correlated errors that affects a least-squares adjustment is the portion of the power spectrum at the frequencies contained in the P matrix. Since we have assumed here that the P matrix varies very slowly, the important frequency components are all near zero frequency. Therefore, using the power spectrum at zero frequency, as the approximation does, is the correct thing to do in this case.

When the above approximation is used, the adjusted parameters and their covariance matrix (computed from the solution using the artificial covariance matrix according to (A.2-1)) are correct within the limitations of the approximation. However, the quadratic form computed from (A.1-25) using the inverse of the artificial covariance matrix for W does not agree with that from (A.1-2) and thus does not have the usual properties described in Section A.1. (Its expected value and degrees of freedom are in general less than $n-m$.) The quadratic form computed from (A.1-2) using the true covariance matrix would be correct, but it would be time-consuming to compute. It is possible to approximate the actual distribution of q from (A.1-25) under this approximation, but that will not be discussed here.

One problem remains. When the covariance matrix of residuals is computed according to (A.1-22) (or the corresponding partitioned form), the true covariance matrix of the observations must be used for S_U, whereas the artificially augmented covariance matrix was used to obtain the covariance matrix of the adjusted observations. Because the covariance matrix of residuals is the difference between the two, if for some observation the variance of the adjusted observation is nearly as large as the variance of the observation, any error in the former caused by the inaccuracy of the approximation will cause a relatively large error in the variance of the residual. The main problem occurs when the conditions of the approximation are not met well, in that the extent of the correlation in observation space exceeds the extent of the similar P matrices. In this case the variance of the adjusted observation will be overestimated, and the computed variance of the residual can actually become negative. The problem can be avoided by using the fact that a reasonable upper limit for the variance of an adjusted observation is as follows:

$$\sigma^2_{u_i} \leq \gamma + \frac{\sigma^2_{u_{c,i}} s_{ii}}{\hat{s}_{ii}} \tag{A.2-2}$$

where $\sigma^2_{u_{c,i}} = P_i S_C P_i^T$ is the computed variance of the adjusted observation, $s_{ii} = \sigma^2_{u_i}$ is the variance of the observation, $\hat{s}_{ii}$ is the augmented variance of the observation according to (A.2-1), and γ is the greatest s_{ij} for $i \neq j$ (largest covariance between this observation and any other). Thus the minimum of $\sigma^2_{u_{c,i}}$ and the limit from (A.2-2) can be used for $\sigma^2_{u_i}$, and the variance of the residual is then obtained by $\sigma^2_{v_i} = \sigma^2_{u_i} - \sigma^2_{u_i}$ instead of by using (A.1-22).

A.3 Variance Adjustment

The solution in Section A.1 assumes that the covariance matrix S_U is known, so that it can be inverted to obtain the weight matrix. Often this is not the case, and some information about it must be obtained from the solution itself. Of course, if nothing at all is known about S_U, there is not much hope. However, if some information is available about it, the solution may be able to estimate the rest by utilizing information contained in the residuals. (An accurate estimate can be obtained only if the number of observations n is sufficiently greater than the number of parameters m so that there is enough information in the residuals. If $n = m$, the residuals are zero.) One example of this was mentioned in Section A.1, concerning the well-known use of the quadratic form to adjust the scale factor of S_U. A more elaborate case is discussed in this section, which is used in the implemented version of the stereo camera model adjustment.

Suppose that the covariance matrix can be expressed as the sum of two positive-definite symmetrical matrices, as follows:

$$S_U = A + \gamma B \qquad \text{(A.3-1)}$$

such that A is known exactly, B is known exactly, and the scale factor γ is unknown except for an a priori value γ_0 and its variance $\sigma^2_{\gamma_0}$. (In the camera model adjustment, A corresponds to errors estimated by the correlator, γ corresponds to the additional error discussed in Section 3.2, and B is its correlation matrix.)

In order to estimate γ, one approach that might be tried is to use the fact that the expected value of the quadratic form is the number of degrees of freedom of the adjustment (the number of observations minus the number of parameters), as mentioned in the previous section. Thus substituting (A.3-1) into (A.1-2) and setting the quadratic form equal to $n-m$ produces the equation $V^T(A+\gamma B)^{-1}V = n-m$. This could be solved for γ. (The residuals would be obtained from a solution using the old value of γ, from the previous iteration.) However, this equation is equivalent to an nth-degree polynomial in γ, and solving it would be very time-consuming. Therefore, a different approach is used.

Let r_i be the ratio of the variance of the ith observation to the variance of the ith residual. Thus

$$r_i = \frac{\sigma^2_{u_i}}{\sigma^2_{v_i}} \qquad \text{(A.3-2)}$$

where $\sigma^2_{v_i}$ is a diagonal element of S_V obtained from (A.1-22). By definition, $\sigma^2_{v_i}$ is the expected value of v_i^2 (since the expected value of v_i is zero). Therefore, using the diagonal elements of (A.3-1) to obtain $\sigma^2_{u_i}$ in (A.3-2) and rearranging produces

$$b_{ii}\gamma = r_i\, \mathcal{E}v_i^2 - a_{ii} \qquad \text{(A.3-3)}$$

where the symbol $\mathcal{E}$ represents the mathematical expectation operator. Now, v_i^2 can be considered to be an estimate of $\mathcal{E}v_i^2$ based on one sample. Thus, if the squared residual (obtained from a solution using the old value of γ from the previous iteration) is used in (A.3-3) in place of its expected value (and r_i from the previous iteration is used), (A.3-3) can be solved for γ. Of course, one sample of a squared residual does not produce a good estimate for the variance, but there are n equations (A.3-3), one for each observation in the adjustment. Thus an adjustment can be done for γ with n observations, using (A.3-3) as the condition equations. These observations will be called "variance observations" to distinguish them from the observations u_i in the main adjustment. (If the covariances were used also, there would be $\frac{1}{2}n(n+1)$ variance observations. However, using this many observations would be time-consuming and would complicate the analysis below, with little added benefit, since the main-diagonal elements contain most of the information except in the case of highly correlated errors. In such a case it may be desirable to perform a rotation to diagonalize A in order to utilize more fully the available information, but this is not done in the implemented stereo camera model adjustment.)

Because the variance of the adjusted observations is usually much smaller than the variance of the observations (assuming that $n>>m$), r_i for most observations is only slightly greater than 1. Thus its effect usually is only a slight correction, and using the value from the previous iteration is satisfactory. In any case, when convergence is achieved, the value will be correct. (In fact, the value of $1/r_i$ averaged over all observations is usually close to $(n-m)/n$, although when the approximation in Section A.2 is used this value must be altered. Thus in many cases it is a reasonable approximation to use a constant value of $n/(n-m)$ for all r_i's.)

In order to combine the above measurements of γ correctly, the covariance matrix of the variance observations must be known in order to obtain the weight matrix. The variance observations according to (A.3-3) correspond to the right side of the equation. Since a_{ii} is known, the covariance matrix of the variance observations is the same as the covariance matrix of $r_i v_i^2$. As an approximation, it is assumed here that r_i is known. Thus, the covariance of the ith and jth variance observations is $r_i r_j$ times the covariance of v_i^2 and v_j^2. In general, the variances and covariances of the squares of variables cannot be obtained if only the variances and covariances of the variables are known. However, if the variables have the normal distribution with zero mean, then the variances and covariances (about the mean) of the squares of the variables are twice the squares of the respective variances and covariances of the variables. Under this assumption, which is valid if the original observations have the normal distribution, the covariance of the ith and jth variance observations is

$$s_{ij} = 2 r_i r_j \sigma^2_{v_i v_j} \tag{A.3-4}$$

where $\sigma^2_{v_i v_j}$ is the i,j element of S_V, which can be obtained by using (A.3-1) and (A.1-22) from the previous iteration. (When $n>>m$, a fair approximation for s_{ij} would be $2\sigma^2_{u_i u_j} = 2(a_{ij}+\gamma b_{ij})^2$, using the value of γ from the previous iteration. This is exact for the diagonal elements but neglects the correlations that the solution has introduced into the adjusted observations. This approximation is used in the camera model adjustment, since it avoids having to compute the entire covariance matrix of residuals.) Then these values of s_{ij} are assembled into the matrix S_Υ, the covariance matrix of the variance observations. (To avoid confusion with the symbols U, W, and P in the main adjustment, the Greek letters Υ, Ω, and Π are used here for the corresponding matrices in the variance adjustment.) Then $\Omega = S_\Upsilon^{-1}$ produces the weight matrix for the variance observations. (It might be pointed out here that the covariance matrix of residuals is singular. However, the matrix composed of the squares of its elements in general is not singular if n is sufficiently large. Even if it is singular, the method in Section A.2 can recover the needed information in many cases.)

Then the variance adjustment can be done using (A.1-17). The Π matrix (corresponding to P there) is a column matrix containing the b_{ii} values, and Υ (corresponding to both U and E in the main adjustment, since (A.3-3) is linear in γ) is a column matrix containing the values $r_i v_i^2 - a_{ii}$. (Do not confuse the symbol Π with the larger symbol used to denote products.) Then

$$\gamma = \frac{\Pi^T \Omega \Upsilon + \dfrac{\gamma_0}{\sigma^2_{\gamma_0}}}{\Pi^T \Omega \Pi + \dfrac{1}{\sigma^2_{\gamma_0}}}$$

$$\sigma^2_\gamma = \frac{1}{\Pi^T \Omega \Pi + \dfrac{1}{\sigma^2_{\gamma_0}}} \qquad \text{(A.3-5)}$$

where the a priori value γ_0 and its weight $1/\sigma^2_{\gamma_0}$ have been introduced in the proper way. (Since Π and Υ each have only one column, the matrix products in (A.3-5) produce 1-by-1 matrices, which are equivalent to scalars.) However, because of random fluctuations it is possible for γ from (A.3-5) to be negative. If this happens, it should be set to zero instead.

Because γ is used in obtaining the weights to be used for computing γ, the above process is iterative. A complete iterative solution for γ could be done on each iteration of the main solution. However, this is not necessary. One iteration of the variance adjustment can be done on each iteration of the main adjustment, and the variance and the main parameters will converge together. Note that, since the main adjustment has not yet converged, it is actually the discrepancies instead of the residuals that are used in Υ in (A.3-5). This will cause γ to be an overestimate on the early iterations. But as the discrepancies converge to the residuals, γ will converge to the proper value.

Instead of using (A.3-5) as is, it can be partitioned in the same manner as (A.1-17) was partitioned to obtain (A.1-23), if the appropriate off-diagonal terms of Ω are negligible. But even if certain off-diagonal terms in the main observation covariance matrix S_U are zero, they won't be zero in S_Υ, because of the correlations introduced by the main adjustment into the residuals. Of course, if the approximation of using twice the squares of the elements of S_U for the elements of S_Υ is used, then the variance solution can be partitioned in the same way as the main solution. In any event, the approximate way of handling correlated errors described in Section A.2 can be used, and this would allow the same partitioning to be used in the two solutions.

A.4 Convergence Acceleration

In Section A.1 it was pointed out that the solution described there using the Gauss method undergoes linear convergence. In many cases this is adequate, but sometimes the convergence is quite slow. This section describes a way of accelerating the convergence of the Gauss method, which converts it into quadratic convergence. This method is used in the stereo camera model adjustment. (In the case where there is only one parameter being adjusted, this method, except for the acceptance test described below, is equivalent to Aitken's extrapolation, described in Acton [1970].)

Let the true (unknown) values of the parameters be represented by the vector G_t, let G_i represent the current values of the parameters used on iteration i (before the correction), and let D_i represent the corrections computed to the parameters by the Gauss method on iteration i. Then ideally $D_i = G_t - G_i$. However, more accurately

$$D_i = A(G_t - G_i) \tag{A.4-1}$$

where A is a constant square matrix. The fact that A differs from the identity matrix causes the linear convergence. (A here corresponds to $I-A$ in the discussion following (A.1-13).) Even more accurately, (A.4-1) would also contain higher-order terms in $G_t - G_i$, which are neglected here. If A could be computed, it could be used to obtain a more accurate correction by solving (A.4-1) for $G_t - G_i$.

Suppose that two different sets of parameter values (corresponding, say, to iterations i and j) are used in the solution and the resulting equations (A.4-1) are differenced. The result is

$$D_i - D_j = A(G_j - G_i) \tag{A.4-2}$$

Everything in this equation is known except A, but it cannot be solved for A because it represents m scalar equations in m^2 unknowns, where m is the number of parameters. However, if m pairs of values are used to obtain m equations (A.4-2), they can be solved. Let C be an m-by-m matrix each of whose columns consists of one $D_i - D_j$ vector, and let B be an m-by-m matrix each of whose columns consists of one $G_j - G_i$ vector. Then the m different equations (A.4-2) are all represented by the one equation

$$C = AB \tag{A.4-3}$$

In the actual procedure, $(D_i - D_j)/s$ and $(G_j - G_i)/s$ are used to form a column of C and B, respectively, where s is the magnitude of the vector $G_j - G_i$ (square root of the sum of squares of its elements). Dividing by s in this way normalizes things to avoid numerical problems caused by the rapidly diminishing size of these vectors during convergence, but the resulting equation (A.4-3) is mathematically equivalent to the above description, since the same factor is applied to corresponding columns of C and B.

Now let $\tilde{D}$ be the desired more accurate vector of corrections. From (A.4-1) it can be seen that $\tilde{D}_i = A^{-1}D_i$. Equation (A.4-3) can be solved to produce $A^{-1} = BC^{-1}$. Substituting the latter into the former produces

$$\tilde{D}_i = BC^{-1}D_i \tag{A.4-4}$$

Adding the results of (A.4-4) to G_i produces the more accurate values of the parameters.

In order for (A.4-4) to be computed, C must be nonsingular. This requires that all of its columns ($D_i - D_j$ pairs) be linearly independent, which requires that $m+1$ different G's be used to obtain the m pairs used in (A.4-2). Different values of G could be chosen deliberately to produce linearly independent columns, but this would require $m+1$ complete computations of D for each iteration, which would defeat the purpose of the acceleration. Instead, values of G and D from $m+1$ successive iterations are used to obtain B and C in (A.4-3).

Therefore, the procedure starts by going through m normal iterations. Then iteration $m+1$ is computed. Values of G and D from iterations 1 through m are each differenced against those of iteration $m+1$ to obtain B and C in (A.4-3). These are used in (A.4-4) to obtain a more accurate correction for iteration $m+1$. Then iteration $m+2$ is computed, and iterations 2 through $m+1$ are each differenced against iteration $m+2$ to obtain an accelerated iteration $m+2$. This process repeats in this manner, always comparing iterations $i-m$ through $i-1$ to iteration i when correcting iteration i, until the convergence tolerance is achieved.

A problem remains, however. As the solution converges, the direction of the error vector $G-G_t$ will tend to approach the constant direction given by the eigenvector corresponding to the largest eigenvalue of $I-A$, with only its magnitude changing. This will result in the columns of B and C becoming nearly proportional to each other, which causes C to become nearly singular. (Indeed, if by accident the error vector started exactly in this direction, C would be singular from the start.) But the eigenvalue of C corresponding to the eigenvector in this direction will not become zero, and it is this eigenvalue that contains the information needed to compute $\tilde{D}$ in this case, since D is in this direction also. In general, those eigenvalues of C that are zero correspond to directions orthogonal to $\tilde{D}$, and thus do not matter. Therefore, even though C may be singular, it contains the needed information. In order to extract this information, some type of generalized inverse might be used instead of the inverse indicated in (A.4-4).

However, in the implemented procedure the following is done to get around this problem. First, if possible, C is inverted in the usual way, and a test is made to determine if there was excessive loss of significance. If the test is passed, the result is used. However, if the matrix was singular or the test was failed, a small positive quantity is added to the diagonal elements of C and another try is made. Adding this quantity to the diagonal elements increases each eigenvalue by this amount. A large eigenvalue will not be appreciably affected by this small change, but a zero eigenvalue will no longer be zero. If all eigenvalues differ from zero by significant amounts, the matrix can be inverted accurately. But there may have been a negative eigenvalue which becomes close to zero by the addition. Therefore, the same test is made on the second try, and if it also fails, another try is made. The implemented procedure subtracts the same quantity from the diagonal elements of the original C matrix and tries again. If this doesn't work either, it gives up and does not accelerate on this iteration. (Using up to m tries with a good matrix inverter would practically guarantee success, because there are only m eigenvalues.) The implemented procedure does these computations in double precision, and the quantity added is 10^{-8}. Since C has been normalized, this changes its largest eigenvalue by roughly one part in 10^8. If eight significant decimal digits were used in the computations, an appropriate quantity to add or subtract would be 10^{-4}.

Before the convergence acceleration computed above is accepted on any particular iteration, one more test needs to be made. If the higher-order terms which were neglected above are large, applying the acceleration might make things worse instead of better. The test that is made involves seeing whether the normal or the accelerated solution is more consistent on two successive iterations. Whichever one was used to produce G on the previous iteration, the other value of G is remembered. Both values of G are obtained for this iteration. Then the magnitude of the difference of the G vectors produced by the normal solution on the two iterations is computed, and the magnitude of the difference of the G vectors produced by the accelerated solution on the two iterations is computed. For this iteration, the solution for which this magnitude is less is accepted. If there was no accelerated solution computed for the previous iteration (because of insufficient iterations or failure of the matrix inversion), the normal solution is used for this iteration. (Thus the accelerated solution will never be used until iteration $m+2$ at the earliest, since the first accelerated solution isn't computed until iteration $m+1$.)

The magnitude of a vector used above is defined as the square root of the sum of the squares of the elements of the vector. In order for this to be a meaningful representation of the distance between two solutions, the elements of the vector should be comparable quantities, with their important effects being the same order of magnitude. If necessary, the actual parameters in the adjustment should be scaled in order to achieve this condition. (This is also desirable to avoid numerical loss of significance.)

If the variance adjustment described in Section A.3 is used, the variance estimate γ can be considered to be one of the parameters for the purposes of convergence acceleration. Then m here corresponds to $m+1$ in the main solution. However, the variance must be scaled appropriately, as described in the previous paragraph.

Because the method described here requires at least $m+2$ iterations in order to accelerate the convergence, it would not be of much use when the number of parameters is large. However, when m is small and convergence is slow, it can be quite useful. The computation time it requires (consisting mostly of an m-by-m matrix inversion) is usually much less than that of the main solution, so a large cost is not paid for its use, even if it turns out not to be needed.

A.5 Automatic Editing

It was mentioned in Section A.1 that the solution there is the maximum-likelihood solution if the errors have the Gaussian (normal) distribution. However, suppose that the errors are from two causes. There are small random errors on every observation approximately normally distributed, called "noise," and there are occasional very large errors, called "wild points." The combined distribution for the total error departs greatly from the normal distribution. It consists of an approximately normal curve added to a function with a very small amplitude but a large width. The use of the unmodified solution in Section A.1 would result in large errors because of the wild points. Some nonlinear solution adapted to the actual total error distribution is needed.

One approach would be to assume a particular total error distribution, and derive the exact maximum-likelihood solution for it. Compared to the ordinary weighted least-squares solution this would have the effect of giving less weight to the points with large errors on the current iteration, since they would lie on the further parts of the error distribution where the curve flattens out instead of following the normal curve. In general this would be quite complicated, and it would add a great deal of nonlinearity to the solution, adversely affecting the convergence. (A crude approximation to this sort of thing is used in the object finder in Chapter 7.)

One reasonable way to approximate the above ideal in many cases can be derived from the following reasoning. Because the amplitude of the probability distribution of the wild points is so small (caused by their infrequence and wide range of values), the total error distribution curve is very nearly the normal curve for small errors (if the noise is normally distributed). But for some value of error the two probability densities are equal, and for errors greater than this the normal curve rapidly becomes negligible, resulting in a flat distribution from there on. Thus the total curve can be approximated by a normal curve out to some threshold value and by a constant beyond there. The maximum-likelihood solution that results from this approximation is to use the points with errors less than the threshold in the usual way and to ignore all other points. Such a process of rejecting outlying points is called "editing."

The correct threshold to use for the editing process depends upon the error distributions. For example, suppose that the wild points occur 10% of the time and have a distribution that is 100 times wider than that of the noise. (Both are assmed to be normally distributed for convenience, but a uniform distribution for the wild points with a width $100\sqrt{2\pi}$ times the standard deviation of the noise would produce the same results.) Then the height of the wild point distribution at the center is only 1/1000 of that of the noise. Thus the two distributions become equal when the noise distribution is at 1/1000 of its peak, since the wild point distribution is practically flat in this region. This occurs at an error of 3.7 standard deviations for the normal curve, and this would be the correct threshold in this case. In practice the exact wild point distribution is seldom known, but using a threshold of three standard deviations for one-dimensional data is usually reasonable and is somewhat customary in editing problems. (Cutting the normal curve off at both sides at three standard deviations results in rejecting only 0.0027 of its area.) The ratio of the threshold to the standard deviation is denoted here by t. (In order to take into account the fact that the standard deviation is not known exactly, it would be better to use a threshold based on Student's t distribution instead of a constant a priori threshold, but if the variance estimate is reasonably accurate this will make little difference.)

Of course, the errors are not known. However, after performing an adjustment the residuals are known, and their covariance matrix can be computed from (A.1-22), with the correction discussed in connection with (A.2-2) imposed when the approximation in Section A.2 is used. Therefore, the editing process used here basically checks to see whether for any observations the absolute value of the residual is greater than t times the standard deviation of the residual. Several refinements are needed to this basic process, however.

Some subsets of observations may be so closely related that, if one of the observations in a subset is wrong because of a wild point, the others probably are wrong also. For example, in the stereo camera model adjustment, if a point seems to be beyond infinity, there are two observations associated with this point, as explained in Chapter 3, and the two observations should be accepted or rejected together because they both came from the same correlator measurement. The optimum way in which to do this is to compute the quadratic form of the vector of residuals for this point with the inverse of its covariance matrix, as follows:

$$q_i = V_i^T[S_{U_i} - P_i S_G P_i^T]^{-1} V_i \tag{A.5-1}$$

(subject to the limit given by (A.2-2) when the approximation in Section A.2 is used), where S_G is the covariance matrix of the adjusted parameters, S_{U_i} is the covariance matrix of the observations in this point, P_i is the matrix of partial derivatives of the observations in this point relative to the parameters, and V_i is the vector of residuals for this point (observations minus adjusted observations). The square root of this quadratic form would correspond to the ratio of the absolute value of a residual to its standard deviation in the case of one observation per point. Thus a limit of t^2 on q_i would produce a cutoff at the same probability density value of the normal curve. However, because of the greater number of dimensions in the space over which the wild points are distributed, their probability density will be less, and thus the value of the threshold t perhaps should be greater than in the one-dimensional case. (In the implemented stereo camera model adjustment, $t = 3$ in the one-dimensional case and $t = 4$ in the two-dimensional case.)

The presence of one wild point may perturb the solution so that it approximately agrees with another wild point. Therefore, a single check for all wild points cannot be completed in one step. After one or more points are rejected, the test must be made again on the remaining points. The most likely candidate for rejection is the observation with the largest ratio of absolute value of residual to standard deviation of residual, or the point with the largest quadratic form from (A.5-1) in the multidimensional generalization. (However, if t is different for different points, this value should be scaled before comparing by dividing the residual by t or the quadratic form by t^2.) As implemented in the stereo camera model adjustment, this point is rejected first if it is beyond the limit. Then the solution is recomputed and the process repeats until no more points seem to need rejecting. Previously rejected points could be retested at each step and reinstated if they are now within the limit, but this is not done in the camera model adjustment. Note that if the basic problem is nonlinear it must be iterated to convergence on each one of these steps so that true residuals will be obtained. Therefore, the editing process consists of outer iterations, each one of which contains the inner iterations of the basic solution.

If the problem is linear and the variance of the observations is known, then the process of comparing a residual to its standard deviation computed from the solution using this observation suffices to indicate whether or not this observation should be rejected. However, if the problem is nonlinear, removing a point from the solution may change things so much that the decision might be different. Also, if the variance is being adjusted (as in Section A.3), the presence of this wild point will cause the variance to be overestimated. Therefore, the residual may be less than t times its overestimated standard deviation but more than t times its true standard deviation. For these reasons, the point with the largest ratio compared to t is tentatively rejected regardless of the size of the ratio, the solution is recomputed (including the variance adjustment) without this point, the residual and its standard deviation are recomputed, and a definite decision on this point is made based on the size of the new ratio. The standard deviation of the residual in this last step must be computed in a different way than usual. Since this observation is not used in the solution, (A.1-22) cannot be used. There will be no correlation between this unused observation and the solution, provided that this observation is not correlated with the observations used in the solution. Therefore, since this residual is the observation minus the adjusted observation computed from the solution, the variance of the residual is the sum of the variance of the observation and the variance of the adjusted observation. For multidimensional observations this generalizes to the sum of the covariance matrices. Thus the quadratic form which is actually compared to t^2 to determine whether a tentatively rejected point should really be rejected is computed as follows (using values from the solution computed without using this point):

$$q_i = V_i^T[S_{U_i} + P_i S_C P_i^T]^{-1} V_i \tag{A.5-2}$$

(The limit given by (A.2-2) can be applied here also, but with an addition instead of a subtraction it is less important.)

The possibility still remains that the presence of many wild points (all about equally bad) may cause such an overestimate of the variance that none of them would be rejected. This possibility can be guarded against in the following way. The computed variance (not including the a priori estimate) is compared to the a priori variance, and, if the ratio is large enough to cause some confidence level to be exceeded, the most suspect point on this outer iteration will not be reinstated yet if it passes the usual test above. Points successively tentatively rejected in this way are accumulated until they fail the usual test, in which case they are rejected, or until the confidence level is no longer exceeded or a given limit on the number of points to remove is reached, in which case they are reinstated. (Thus if the solution does not reach a set of retained points that indicate that the rejected points are actually bad, the likelihood is that by chance the confidence level was exceeded with good data, and the points should be reinstated. An earlier form of the stereo camera model adjustment reported in Gennery [1977] did not include this last step and thus ran the risk of once in a while rejecting many good points.) The implemented stereo camera model adjustment uses an F test for this purpose, with a confidence level of 0.98, although the presence of the two components of error according to (A.3-1) makes this nonrigorous.

An additional explanation perhaps is in order concerning one matter. Suppose that there is a wild point with no other points in the same region of observation space and that the nature of the problem is such that this point thereby forces the solution into near agreement with it. (For example, consider the simple case of fitting a linear one-dimensional function to some data. If most of the points are clustered in a fairly narrow interval of the independent variable, but there is one point at a distant value of the independent variable with an erroneous value of the dependent variable, this one wild point will tilt the straight-line fit so that it nearly passes through this point.) This wild point will have a very small residual when it is used in the solution, and thus it might appear that it would not be the prime candidate for rejection. However, in such a case almost all of the information in the adjusted value of this observation is coming from this observation itself, and thus the variance of this adjusted observation is nearly as great as the variance of the observation. Since the variance of the residual is the difference of these quantities, it will be very small. As a result, it turns out that, even though the residual is small, its standard deviation is even smaller. Therefore, taking the ratio of these quantities (or using the more general result from (A.5-1)) identifies this point as the one to be removed.

The basic idea of examining the residuals for editing purposes is fairly common. (See, for example, Davis [1967].) However, the method described above contains some refinements, such as the use of the *F* test.

Appendix B

STEREO CAMERA MODEL

In this appendix the particular set of parameters that constitute the stereo camera model used in this work is defined, and a method is described for computing the quantities needed in Chapters 3, 4, and 5 that are functions of these parameters.

The stereo camera model might most reasonably be defined to consist of six parameters defining the relative position and orientation of the two cameras. Many different sets of six quantities are possible; those that are used in the present work are described below. However, the magnitude of the distance between the cameras is sometimes considered separately (because it cannot be determined by the self-calibration method described in Chapter 3), leaving five quantities in the camera model proper. In addition, a scale factor for the pictures, related to the principal distance or focal length, may be included here (and can be adjusted in the same self-calibration procedure, although it usually is better adjusted with the distortion calibration for the individual cameras). There may be separate scale factors for each picture or a single one for both. Therefore, the total number of parameters considered to constitute the stereo camera model may be five, six, seven, or eight. The implemented version of the stereo camera model self-calibration adjusts for only the basic five parameters, although the necessary information is included in this appendix to enable the principal distances to be included in the adjustment also.

If a full set of six parameters defining the relative position and orientation were to be considered to constitute the stereo camera model, a reasonable choice for the parameters might be the three Cartesian components of the vector from Camera 1 to Camera 2 and three angles defining the orientation of Camera 2. These all would be expressed in the Camera 1 coordinate system, since we are concerned here only with relative (not absolute) position and orientation. However, since the magnitude of the vector between cameras is considered separately here, only the direction of the unit vector pointing towards Camera 2 is considered, which can be specified by two quantities. Depending on what two quantities are chosen, a degeneracy occurs in some position. Here, the direction of the unit vector is specified by an azimuth angle and an elevation angle, as in Hannah [1974]. The degenerate position then occurs when one camera is directly above the other, a situation not usually encountered in stereo work and one which can be defined away by rotating the Camera 1 coordinate system about its principal axis. These azimuth and elevation angles, and the pan, tilt, and roll angles (also as in Hannah [1974]) which specify the orientation of Camera 2 are the five quantities which constitute the stereo camera model that is adjusted in Chapter 3. However, the two principal distances are also used in the following computations and could be considered to be camera model parameters. The magnitude of the vector from Camera 1 to Camera 2 does not enter into the computations in this appendix, but is used in the computations in Chapters 4 and 5.

Definitions of the above quantities and others will now be given. The picture-taking process in each camera is idealized as a central projection from the real world onto an image plane perpendicular to the lens axis at a distance f_1 or f_2 (for Camera 1 or Camera 2, respectively) in front of the center of projection. (The quantity f_1 or f_2 is sometimes referred to as "focal length," which is not the correct term if the camera is not focused at infinity. The term "principal distance" is also used, and it will be used here for want of a better term. The center of projection is often called the "lens center," which is correct only in the thin-lens approximation. For thick lenses it is actually the primary principal point.) Each camera has a Cartesian coordinate system with the origin at the center of projection, x to the right in the image plane, y up in the image plane, and z outwards along the lens axis. Thus the coordinate system is left-handed. Measured values of x and y for a corresponding point in the two image planes will have a subscript 1 or 2 to denote Camera 1 or 2, respectively. The azimuth and elevation of the Camera 2 origin relative to the Camera 1 coordinate system are denoted by α_1 and α_2 (positive to the right from the z axis and up), respectively. The pan, tilt, and roll of the Camera 2 coordinate system relative to the Camera 1 coordinate system are denoted by β_1, β_2, and β_3, (positive right, up, and right), respectively.

If the ray from the Camera 1 origin through the point x_1, y_1 in the Camera 1 image plane is back-projected into the Camera 2 image plane, a line segment is produced. Let x_0 and y_0 denote the Camera 2 image-plane coordinates of the end point of this line segment (corresponding to a point at an infinite distance on the ray), and let c_x and c_y denote the direction cosines of the line segment (in the direction away from x_0, y_0) relative to the x_2 and y_2 axes, respectively. Then the problem at hand is to use the quantities x_1 and y_1 and the camera model parameters previously defined to compute x_0, y_0, c_x, and c_y. Also needed for the computations in Chapter 5 (and needed in order to compute the above quantities) are the unit vector $\mathbf{1}_r$ pointing from the Camera 1 origin to the Camera 2 origin (in Camera 1 coordinates), and the rotation matrix B for transforming Camera 1 coordinates into Camera 2 coordinates, which are functions of the camera model parameters only. The partial derivatives of all of these quantities with respect to the camera model parameters are also needed.

Two vectors that will be needed later are defined as follows:

$$\mathbf{p} = \begin{bmatrix} x_1 \\ y_1 \\ f_1 \end{bmatrix} \qquad \mathbf{1}_z = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{B-1}$$

The first step in deriving the needed mathematics consists of defining the rotation matrices associated with the angles α_1, α_2, β_1, β_2, and β_3. Notice that A_1 and A_2 are defined with the opposite direction of rotation from B_1 and B_2. This is because the A's will be used to rotate a vector whereas the B's will be used to rotate the coordinate system.

$$A_1 = \begin{bmatrix} \cos\alpha_1 & 0 & \sin\alpha_1 \\ 0 & 1 & 0 \\ -\sin\alpha_1 & 0 & \cos\alpha_1 \end{bmatrix} \qquad \frac{dA_1}{d\alpha_1} = \begin{bmatrix} -\sin\alpha_1 & 0 & \cos\alpha_1 \\ 0 & 0 & 0 \\ -\cos\alpha_1 & 0 & -\sin\alpha_1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_2 & \sin\alpha_2 \\ 0 & -\sin\alpha_2 & \cos\alpha_2 \end{bmatrix} \qquad \frac{dA_2}{d\alpha_2} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\sin\alpha_2 & \cos\alpha_2 \\ 0 & -\cos\alpha_2 & -\sin\alpha_2 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} \cos\beta_1 & 0 & -\sin\beta_1 \\ 0 & 1 & 0 \\ \sin\beta_1 & 0 & \cos\beta_1 \end{bmatrix} \qquad \frac{dB_1}{d\beta_1} = \begin{bmatrix} -\sin\beta_1 & 0 & -\cos\beta_1 \\ 0 & 0 & 0 \\ \cos\beta_1 & 0 & -\sin\beta_1 \end{bmatrix} \qquad \text{(B-2)}$$

$$B_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta_2 & -\sin\beta_2 \\ 0 & \sin\beta_2 & \cos\beta_2 \end{bmatrix} \qquad \frac{dB_2}{d\beta_2} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\sin\beta_2 & -\cos\beta_2 \\ 0 & \cos\beta_2 & -\sin\beta_2 \end{bmatrix}$$

$$B_3 = \begin{bmatrix} \cos\beta_3 & -\sin\beta_3 & 0 \\ \sin\beta_3 & \cos\beta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \frac{dB_3}{d\beta_3} = \begin{bmatrix} -\sin\beta_3 & -\cos\beta_3 & 0 \\ \cos\beta_3 & -\sin\beta_3 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The unit vector pointing from the Camera 1 origin to the Camera 2 origin is just the unit z vector rotated through the elevation and azimuth angles:

$$\mathbf{1}_r = A_1 A_2 \mathbf{1}_z \qquad \text{(B-3)}$$

To convert a vector from being expressed in the Camera 1 coordinate system to being expressed in the Camera 2 coordinate system, the coordinate system must be rotated through the pan, tilt, and roll angles (in addition to being translated). Thus the rotation matrix by which the vector must be premultiplied is

$$B = B_3 B_2 B_1 \qquad \text{(B-4)}$$

The partial derivatives of $\mathbf{1}_r$ and B with respect to the camera model parameters are as follows:

$$
\frac{\partial \mathbf{1}_r}{\partial \alpha_1} = \frac{dA_1}{d\alpha_1} A_2 \mathbf{1}_z \qquad \frac{\partial \mathbf{1}_r}{\partial \alpha_2} = A_1 \frac{dA_2}{d\alpha_2} \mathbf{1}_z
$$

$$
\frac{\partial B}{\partial \beta_1} = B_3 B_2 \frac{dB_1}{d\beta_1} \qquad \frac{\partial B}{\partial \beta_2} = B_3 \frac{dB_2}{d\beta_2} B_1 \qquad \frac{\partial B}{\partial \beta_3} = \frac{dB_3}{d\beta_3} B_2 B_1 \tag{B-5}
$$

with all others equal to zero.

Now the infinity point x_0, y_0 will be derived. An image point in the Camera 1 image plane has a three-dimensional position in the Camera 1 coordinate system given by the vector $\mathbf{p} = [x_1 \; y_1 \; f_1]^{\mathrm{T}}$. Since we are concerned at the moment about the infinity point we can ignore the translation between the camera coordinate systems and consider only the rotation. To express the vector $\mathbf{p}$ in a coordinate system aligned with Camera 2 the coordinate system is rotated by premultiplying by the B matrix defined above. Let the resulting vector be denoted by $\mathbf{u}$. Thus

$$
\begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = B\mathbf{p} \tag{B-6}
$$

The projection of the point given by the above vector into the Camera 2 image plane is given by a vector in the same direction as the above vector but having a z component equal to f_2. Therefore,

$$
x_0 = \frac{f_2 u_x}{u_z} \qquad y_0 = \frac{f_2 u_y}{u_z} \tag{B-7}
$$

The partial derivatives of $\mathbf{u}$ with respect to β_1, β_2, and β_3 can be obtained by replacing B in (B-6) by the corresponding derivatives of B from (B-5). If the partial derivatives with respect to f_1 are desired, they can be obtained by replacing $\mathbf{p}$ by $\mathbf{1}_z$ in (B-6), since $\partial \mathbf{p} / \partial f_1 = \mathbf{1}_z$. Equations (B-7) then can be differentiated to obtain the partial derivatives of x_0 and y_0, as follows:

$$\frac{\partial x_0}{\partial g} = \frac{f_2}{u_z}\cdot\frac{\partial u_x}{\partial g} - \frac{f_2 u_x}{u_z^2}\cdot\frac{\partial u_z}{\partial g}$$

$$\frac{\partial y_0}{\partial g} = \frac{f_2}{u_z}\cdot\frac{\partial u_y}{\partial g} - \frac{f_2 u_y}{u_z^2}\cdot\frac{\partial u_z}{\partial g}$$

$$\frac{\partial x_0}{\partial f_2} = \frac{u_x}{u_z} \tag{B-8}$$

$$\frac{\partial y_0}{\partial f_2} = \frac{u_y}{u_z}$$

where g denotes β_1, β_2, β_3, or f_1. (The partial derivatives of x_0 and y_0 with respect to α_1 and α_2 are zero.)

The point x_0, y_0 is the end of the desired line segment. The direction cosines c_x and c_y can be found by using the fact that the desired line is the intersection of the Camera 2 image plane with the plane defined by the Camera 2 center of projection and the ray corresponding to the Camera 1 image point x_1, y_1.

Thus we proceed as follows. The ray which corresponds to the image point x_1, y_1 in the Camera 1 image plane is given by the direction of the vector $\mathbf{p} = [x_1\ y_1\ f_1]^T$, in Camera 1 coordinates. First we must determine the plane containing this ray and the Camera 2 center of projection. The normal to this plane is given by the direction of the vector cross product of $\mathbf{p}$ and the vector $\mathbf{1}_r$ from (B-3) giving the direction of the Camera 2 center of projection from Camera 1 center of projection. Therefore, the normal to the desired plane is $\mathbf{p} \times \mathbf{1}_r$, in Camera 1 coordinates. To express this normal in Camera 2 coordinates we must rotate the coordinate system by the pan, tilt, and roll angles. The result is $B(\mathbf{p} \times \mathbf{1}_r)$. The normal to the Camera 2 image plane in Camera 2 coordinates is $\mathbf{1}_z$. The vector along the intersection of these two planes is the cross product of the normals to the two planes, namely $\mathbf{1}_z \times B(\mathbf{p} \times \mathbf{1}_r)$. This is the desired line which is the projection of the ray into the Camera 2 image plane, expressed in Camera 2 coordinates, and thus its x and y components are proportional to the desired direction cosines. Since the vector lies in the Camera 2 image plane, its z component is zero. Thus, if we call this vector $\mathbf{v}$, we have

$$\begin{bmatrix} v_x \\ v_y \\ 0 \end{bmatrix} = \mathbf{1}_z \times B(\mathbf{p} \times \mathbf{1}_r) \tag{B-9}$$

Application of either the right-hand rule or the left-hand rule consistently to the above two cross products will verify that the above vector has the correct polarity, that is, it points away from x_0, y_0 along the line segment. The direction cosines c_x and c_y can now be computed simply as follows from the results of (B-9):

$$c_x = \frac{v_x}{\sqrt{v_x^2 + v_y^2}}$$

$$c_y = \frac{v_y}{\sqrt{v_x^2 + v_y^2}} \qquad \text{(B-10)}$$

The partial derivatives of $\mathbf{v}$ with respect to the α's and β's can be obtained by replacing in turn $\mathbf{1}_r$ and B in (B-9) by the corresponding derivatives from (B-5). The partial derivatives with respect to f_1 can be obtained by replacing $\mathbf{p}$ in (B-9) by $\mathbf{1}_z$. Then the partial derivatives of c_x and c_y are obtained as follows, where g denotes any of the parameters (α's, β's, or f_1):

$$\frac{\partial c_x}{\partial g} = \frac{v_y^2 \frac{\partial v_x}{\partial g} - v_x v_y \frac{\partial v_y}{\partial g}}{(v_x^2 + v_y^2)^{3/2}}$$

$$\frac{\partial c_y}{\partial g} = \frac{v_x^2 \frac{\partial v_y}{\partial g} - v_x v_y \frac{\partial v_x}{\partial g}}{(v_x^2 + v_y^2)^{3/2}} \qquad \text{(B-11)}$$

The results are $\mathbf{1}_r$ and B for a given camera model; x_0, y_0, c_x, and c_y for a given point and a given camera model; and the partial derivatives of these quantities.

The above computations were expressed in terms of matrices and vectors as much as possible, so that the partial derivatives were easy to obtain. In the implemented computer program the matrix operations are performed numerically by standard procedures. Therefore, there is no need to expand these equations to scalar form analytically, except in a few cases where considerable computation time can be saved. In particular, the product of A_1A_2 times $\mathbf{1}_z$ reduces to just taking the third column of A_1A_2. The cross products are written out in the code for the program; this reduces the cross product of $\mathbf{1}_z$ times another vector to just picking two appropriate terms of the vector, with an appropriate sign change.

Appendix C

MARS PICTURES

The Mars pictures used in this research were extracted from a pair of large mosaics produced by the Viking Lander Imaging Team from pictures taken by the two cameras on the Viking Lander 1. The pictures form a stereo pair of the Martian landscape in front of the lander, covering about 173° in azimuth and about 66° in elevation. However, much of these pictures do not contain corresponding areas in the two pictures because of occlusion by parts of the lander. Also, the portions in the extreme distance probably would not allow accurate information to be obtained about small objects such as rocks. A suitable portion was chosen to test the methods in this thesis, consisting of an area about 16° in elevation by 20° in azimuth in the left picture and 18° in elevation by 28° in azimuth in the right picture. Smaller portions of these were used to generate the examples in this thesis, each about 10° by 10°. These are shown in Figure C-1.

The brightness value of each pixel in the pictures is represented by an eight-bit integer. The pixel spacing of the pictures is 0.04° in azimuth and elevation. (Azimuth and elevation form a spherical coordinate system. Therefore, the central angle subtended by a one-pixel shift in azimuth is 0.04° times the cosine of the elevation angle.) The two cameras are 0.8187 meters apart. The height of the cameras is 1.3 meters above the reference plane (nominal ground surface).

The principal noise source in the pictures supposedly is shot noise from the photodiode sensor. This causes the standard deviation to be proportional to the square root of the pixel values. In order to produce a constant standard deviation, the square root of each pixel value was taken and the result was multiplied by 16 to rescale it to be suitable for an eight-bit picture. The standard deviation of the noise in the resulting pictures was estimated to be about 3. These modified pictures were used by the programs described in this thesis. However, to produce the figures shown herein, the original pictures were changed by a different nonlinear function to enhance their contrast, in order to compensate partially for the inadequacies of the printing device.

Each picture shown in Figure C-1 is 256 pixels by 256 pixels. The azimuth and elevation from the left camera to the center of the picture are about 18° and -20°, respectively, relative to the perpendicular to the camera baseline and relative to the reference plane. The distances to the points in the scene range from about 3 meters to about 4.5 meters.

The white blob in the left picture is an out-of-focus part of the lander's arm, which was present when this part of the mosaic was taken but was in a different position for other portions and for the other picture. It represents erroneous data to the stereo program.

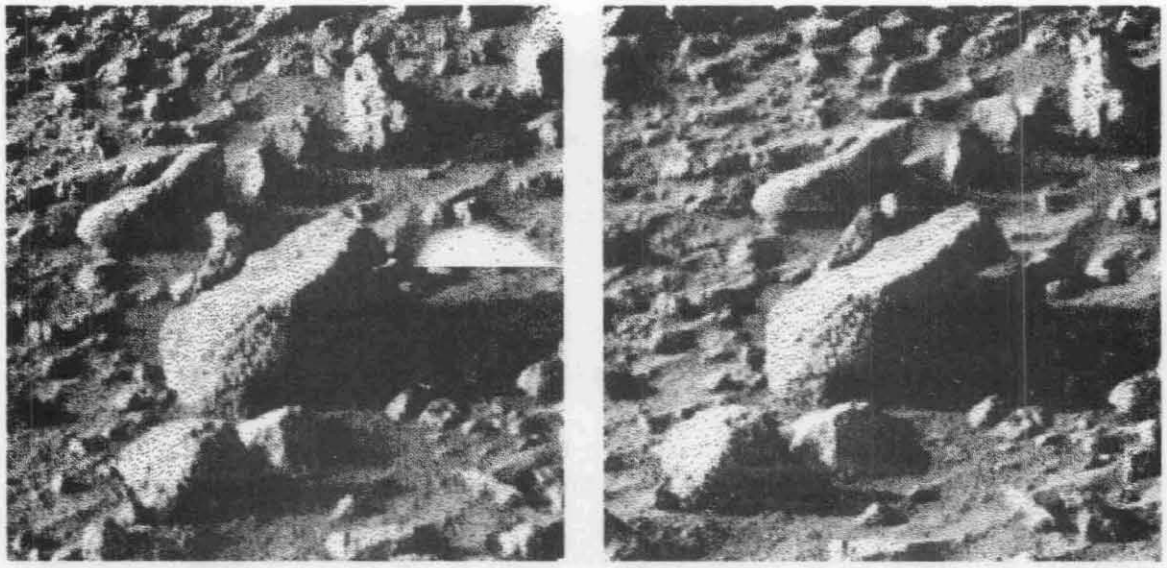

Figure C-1. Stereo pair of Martian surface.

BIBLIOGRAPHY

Acton, F.S. [1970]. *Numerical Methods that Work*, Harper & Row.

Agin, G.J., and Binford, T.O. [1973]. "Computer Description of Curved Objects," *Proceedings of the Third International Joint Conference on Artificial Intelligence*, Stanford, Calif., August 1973, pp. 629-640.

Arnold, R.D. [1978]. "Local Context in Matching Edges for Stereo Vision," *Proceedings: Image Understanding Workshop* (Defense Advanced Research Projects Agency), Cambridge, Mass., May 1978 (Science Applications, Inc., Report Number SAI-79-749-WA), pp. 65-72.

Bard, Y. [1974]. *Nonlinear Parameter Estimation*, Academic Press.

Baumgart, B.G. [1974]. "Geometric Modeling for Computer Vision," Stanford Artificial Intelligence Laboratory Memo AIM-249, Stanford University, Oct. 1974.

Blackman, R.B., and Tukey, J.W. [1958]. *The Measurement of Power Spectra*, Dover Publications.

Bolles, R.C. [1976]. "Verification Vision within a Programmable Assembly System," Stanford Artificial Intelligence Laboratory Memo AIM-295, Stanford University, Dec. 1976.

Brown, D.C. [1955]. "A Matrix Treatment of the General Problem of Least Squares Considering Correlated Obseravations", Report No 937, Ballistic Research Laboratories, Aberdeen Md., May 1955.

Brown, D.C. [1957]. "A Treatment of Analytical Photogrammetry with Emphasis on Ballistic Camera Applications" (Appendix A, "A Treatment of the General Problem of Least Squares and the Associated Error Propagation"), RCA Data Reduction Technical Report No. 39 (AFMTC-TR-57-22), Patrick AFB, Florida, August 1957.

Davis, R.G. [1967]. "Advanced Techniques for the Rigorous Analytical Adjustment of Large Photogrammetric Nets," *Photogrammetria*, Vol. 22, pp. 191-205.

Duda, R., and Hart, P. [1973]. *Pattern Recognition and Scene Analysis*, Wiley.

Forsythe, G.E., and Moler, C.B. [1967]. *Computer Solution of Linear Algebraic Systems*, Prentice-Hall.

Ganapathy, S. [1975]. "Reconstruction of Scenes Containing Polyhedra from Stereo Pairs of Views," Stanford Artificial Intelligence Laboratory Memo AIM-272, Stanford University, Dec. 1975.

Gennery, D.B. [1966]. "Direct Digital Filters for General-Purpose Use", ETR-TR-66-2 (RCA MTP Math Services Technical Report No. 82), Patrick AFB, Florida, Jan. 1966.

Gennery, D.B. [1977]. "A Stereo Vision System for an Autonomous Vehicle," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Cambridge, Mass., August 1977, pp. 576-582.

Gennery, D.B. [1979]. "Object Detection and Measurement Using Stereo Vision", *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, Tokyo, August 1979, pp. 320-327.

Graybill, F.A. [1961]. *An Introduction to Linear Statistical Models*, Volume I, McGraw-Hill Book Company.

Grenander, U. [1954]. "On the Estimation of Regression Coefficients in the Case of an Autocorrelated Disturbance," *Annals of Mathematical Statitistics*, Vol. 25, pp. 252-272.

Hannah, M.J. [1974]. "Computer Matching of Areas in Stereo Images," Stanford Artificial Intelligence Laboratory Memo AIM-239, Stanford University, July 1974.

Hanson, A.R., and Riseman, E.R. (eds.) [1978]. *Computer Vision Systems*, Academic Press.

Hogg, R.V., and Craig, A.T. [1965]. *Introduction to Mathematical Statistics* (Second Edition), The Macmillan Company.

Hohn, F.E. [1973]. *Elementary Matrix Algebra* (Third Edition), The MacMillan Company.

Kalman, R.E. [1960]. "A New Approach to Linear Filtering and Prediction Problems", *Journal of Basic Engineering*.

Levine, M.D., O'Handley, D.A., and Yagi, G.M. [1973]. "Computer Determination of Depth Maps," *Computer Graphics and Image Processing*, Vol. 2, pp. 131-150.

Lewis, R.A., and Johnston, A.R. [1977]. "A Scanning Laser Rangefinder for a Robotic Vehicle," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Cambridge, Mass., August 1977, pp 762-768.

Marr, D., and Poggio, T. [1976]. "Cooperative Computation of Stereo Disparity," *Science*, Vol. 194, pp. 283-287.

Mikhail, E.M. (with contributions by F. Ackermann) [1976]. *Observations and Least Squares*, IEP (Thomas Y. Crowell Company).

Milgram, D., and Bjorklund, C. [1979]. "Superposition," Lockheed Missiles and Space Company internal memo, Palo Alto, Calif.

Moravec, H.P. [1977]. "Towards Automatic Visual Obstacle Avoidance," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Cambridge, Mass., August 1977, p. 584.

Moravec, H.P. [1979]. "Visual Mapping by a Robot Rover", *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, Tokyo, August 1979, pp. 598–600.

Moravec, H.P. [1980]. "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," Ph.D. Dissertation, Stanford University.

Mori, K., Kidode, M., and Asada, H. [1973]. "An iterative Prediction and Correction Method for Automatic Stereocomparison," *Computer Graphics and Image Processing*, Vol. 2, pp. 393–401.

Nevatia, R., and Binford, T.O. [1977]. "Description and Recognition of Curved Objects," *Artificial Intelligence*, Vol. 8, pp. 77-98.

O'Handley, D.A. [1973]. "Scene Analysis in Support of a Mars Rover," *Computer Graphics and Image Processing*, Vol. 2, pp. 281-297.

Price, K. [1978]. "Symbolic Matching and Analysis with Substantial Changes in Orientation," *Proceedings: Image Understanding Workshop*, (Defense Advanced Research Projects Agency), Cambridge, Mass., May 1978 (Science Applications, Inc., Report Number SAI-79-749-WA), pp. 93-99.

Quam, L.H. [1968]. "Computer Comparison of Pictures," Stanford Artificial Intelligence Laboratory Memo 144, Stanford University.

Schultz, M.H. [1973]. *Spline Analysis*, Prentice-Hall.

Schut, G.H. [1957]. "An Analysis of Methods and Results in Analytical Aerial Triangulation," *Photogrammetria*, Vol. 14, pp. 16–33.

Schut, G.H. [1959]. "Remarks on the Theory of Analytical Triangulation," *Photogrammetria*, Vol. 16, pp. 57-66.

Shirai, Y. [1978]. "Recent Advances in 3-D Scene Analysis," *Proceedings of the Fourth International Joint Conference on Pattern Recognition*, Kyoto, Japan, Nov. 1978.

Sobel, I. [1970]. "Camera Models and Machine Perception," Stanford Artificial Intelligence Laboratory Memo AIM-121, Stanford University.

Thompson, A.M. [1977]. "The Navigation System of the JPL Robot," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Cambridge, Mass., August 1977, pp 749-757.

Thompson, C. [1975]. "Depth Perception in Stereo Computer Vision," Stanford Artificial Intelligence Laboratory Memo AIM-268, Stanford University, Oct. 1975.

Ullman, S. [1976]. "The Interpretation of Structure from Motion," Artificial Intelligence Memo 476, Massachusetts Institute of Technology, Oct. 1976.

Watson, G.S. [1967]. "Linear Least Squares Regression," *Annals of Mathematical Statistics*, Vol. 38, pp. 1679-1699.

Yakimovsky, Y. and Cunningham, R. [1978]. "A System for Extracting Three-Dimensional Measurements from a Stereo Pair of TV Cameras," *Computer Graphics and Image Processing*, Vol. 7, pp. 195-210.

Zucker, S.W. [1976]. "Relaxation Labelling and the Reduction of Local Ambiguities," *Proceedings of the Third International Joint Conference on Pattern Recognition*, San Diego, Calif., November 1976, pp. 852-861.