PROCEEDINGS:
IMAGE UNDERSTANDING WORKSHOP

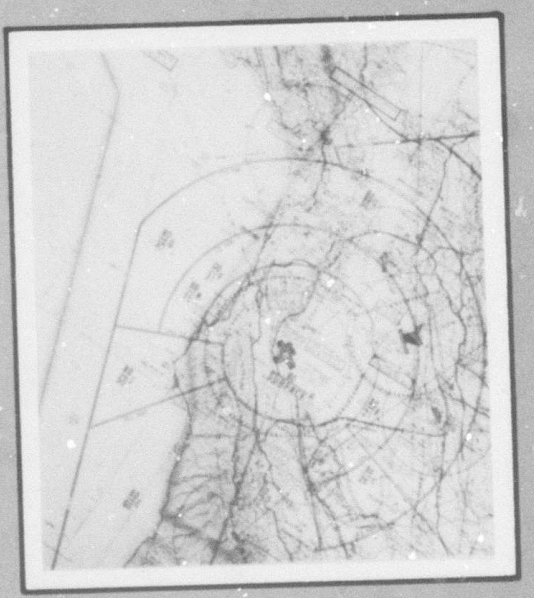
(12)

LEVEL III

APRIL 1979

Sponsored by:
Information Processing Techniques Office
Defense Advanced Research Projects Agency

Hosted by:
SRI International
Menlo Park, California



Aerial and Map Imagery of San Francisco Bay Area

Science Applications, Inc.

LEVEL II (12)

(6) # IMAGE UNDERSTANDING.

Proceedings of a Workshop
Held at
Pala Alto, California.
April 24-25, 1979.

Sponsored by the
Defense Advanced Research Projects Agency

(11) Apr 79 (12) 284p.

(9) Semiannual technical rept.
Nov 78 – Apr 79

(14)
Science Applications, Inc.
Report Number SAI-80-895-WA
(10) Lee S. Baumann
Workshop Organizer and
Proceedings Editor

This report was supported by
The Defense Advanced Research
Projects Agency under DARPA
Order No. 3456, contract No. MDA903-78-C-0095
(15) monitored by the
Defense Supply Service, Washington, D.C.

w DARPA Order-3456

407 154

LB

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: SAI-80-895-WA	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Proceedings: Image Understanding Workshop, April 1979 A064765	5. TYPE OF REPORT & PERIOD COVERED: Semiannual Technical November 1978-April 1979 / 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Lee S. Baumann (Ed.)	8. CONTRACT OR GRANT NUMBER(s): MDA903-78-C-0095
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Science Applications, Inc. 1911 N. Fort Myer Drive, Suite 1200 Arlington, Virginia 22209	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: DARPA Order No. 3456
11. CONTROLLING OFFICE NAME AND ADDRESS: Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209	12. REPORT DATE: April 1979 / 13. NUMBER OF PAGES: 174
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for 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)

Digital Image Processing, Image Understanding, Scene Analysis, Edge Detection, Image Segmentation, CCD Arrays, CCD Processors

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

This document contains the technical papers and outlines of semi-annual progress reports presented by the research activities in Image Understanding sponsored by the Information Processing Techniques Office; Defense Advanced Research Projects Agency. The papers were presented at a workshop conducted on 24-25 April 1979 in Palo Alto, California.

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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

* Not Presented

FORWARD

The Ninth Image Understanding Workshop was conducted under the auspices of Major Larry E. Druffel, United States Air Forces, program manager for the I.U. Research Project in the Information Processing Techniques Office, Defense Advanced Research Projects Agency. As outlined at the Eighth I.U. Workshop, held at Pittsburgh, Pennsylvania, some six months ago, Major Druffel has diligently worked toward a narrowing of focus in the program as the time approaches for the planned concept demonstration which it is hoped will be the capstone to this five year research effort. As previously noted, the intended focus is suggested as "the development of the tools needed for inclusion in some future system", and at this workshop the formulation of a scenario which is to provide a context for the concept demonstration was discussed in an effort to move toward an agreed upon program. It was also noted that although the plan is to move toward a concept demonstration in the next two or three years, the I.U. Program has not lost sight of the need for fundamental research which is required to support future capabilities.

Major Druffel has stated his appreciation for the effort put forth by all of the government research and user personnel in attending these periodic workshops and helping to provide guidance to the ARPA research community. Furthermore, many of the attendees have indicated their recognition of the value gained by the entire research community through the interaction and cross fertilization provided by the various researchers and the diversified user community both in the Image Understanding and in other related research programs.

These proceedings contain the program reviews presented by the Principal Investigators and Technical Reports prepared by selected research personnel at the Ninth Image Understanding Workshop held at Palo Alto, California on 24-25 April 1979. In attendance at the workshop, in addition to the University and Industrial Research Personnel, were representatives from many Army, Navy, Air Force and Government Agency Organizations interested in the accomplishments of this research program. As usual, the workshop provided for a lively exchange of views between the potential user community and those organizations active in the I.U. Research Program. In addition, a panel discussion was conducted between several government research organizations concerning a plan for utilization of emerging technology in Image Understanding by the Defense Mapping Agency. Also, participants were afforded the opportunity to visit the Artificial Intelligence Center at SRI International, the Artificial Intelligence Laboratory at Stanford University, and the Palo Alto Research Laboratory of the Lockheed Missiles and Space Company.

The workshop was hosted by Dr. Martin A. Fischler, senior computer scientist at the Artificial Intelligence Center of SRI International. The workshop organizer wishes to thank Dr. Fischler for his efforts at making the workshop a success and also to recognize the efforts of Miss Jean Burnet, Manager of Conference Services at SRI International, for her excellent advice and assistance. Appreciation is also due Miss Carrie Howell of Science Applications, Incorporated for providing typing support for mailings and the collection and arrangement of the conference proceedings as well as on-site assistance during the conference. Typing assistance was also provided by Miss Jackqueline Frye of the SAI staff.

The cover design was created by Mr. Marco Fillipini of the Art Department of Science Applications, Inc. from material supplied by Dr. Martin Fischler of SRI International. The map data and aerial photography shown are representative of that used by the SRI International research group in Image Understanding on several of their on-going projects. Dr. Fischler states that caption should read "using Map Knowledge to Interpret Aerial Imagery", an important step in the SRI International Image recognition process. An in-depth description of the SRI International Program is contained in this volume as well as in the previous I.U. Workshop Proceedings.

Lee S. Baumann
Science Applications, Inc.
Workshop Organizer

AUTHOR INDEX

SESSION I

PROGRAM REVIEWS

BY

PRINCIPAL INVESTIGATORS

THE SRI IMAGE UNDERSTANDING PROGRAM

M.A. Fischler (Principal Investigator),
O.J. Agin, H.G. Barrow, R.C. Bolles,
L.H. Quam, J.M. Tenenbaum, H.C. Wolf
SRI International
Menlo Park, California

INTRODUCTION

Research at SRI International under the ARPA Image Understanding Program was initiated to investigate ways in which diverse sources of knowledge might be brought to bear on the problem of analyzing and interpreting aerial images. The initial phase of research was exploratory and identified various means for exploiting knowledge in processing aerial photographs for such military applications as cartography, intelligence, weapon guidance, and targeting. A key concept is the use of a generalized digital map to guide the process of image analysis.

The results of this earlier work were integrated in an interactive computer system called "Hawkeye" [1]. This system provides necessary basic facilities for a wide range of tasks and a framework within which specialist programs can be integrated.

Research is now focused on the development of a program capable of expert performance in a specific task domain: road monitoring. The following sections of this paper present an overview as well as some recent technical results produced in this ongoing effort.

OBJECTIVE

The primary objective of this research is to build a computer system that "understands" the nature of roads and road events. It should be capable of performing such tasks as:

(1) Finding roads in aerial imagery

(2) Distinguishing vehicles on roads from shadows, signposts, road markings, etc.

(3) Comparing multiple images and symbolic information pertaining to the same road segment, and deciding whether significant changes have occurred.

The system should be capable of performing the above tasks even when the roads are partially occluded by clouds or terrain features, or are viewed from arbitrary angles and distances, or pass through a variety of terrains.

APPROACH

To achieve the above capabilities, we are developing two "expert" subsystems: the "Road Expert" and the "Vehicle Expert." The Road Expert knows mainly about roads, how to find them in imagery, and what things belong on them. It works at low-to-intermediate resolution (e.g., from 1 to 20 feet of ground distance per image pixel) and has the ability to distinguish vehicles from other road detail. The Vehicle Expert works on higher-resolution imagery and can identify vehicles as to type. We are concentrating our efforts on the Road Expert and therefore will limit most of our discussion here to this component of our system.

The major tasks automatically performed by the Road Expert are:

(1) Image/map correspondence--Place a newly acquired image into geographic correspondence with the map data base.

(2) Road tracking--Precisely mark the center line of selected visible sections of road in the image.

(3) Anomaly analysis--Locate and analyze anomalous objects on, and adjacent to, the road surface; identify potential vehicles.

The image/map correspondence task is accomplished by locating roads and road features as landmarks; correspondence is performed at resolutions as coarse as 20 feet/pixel so that a reasonably wide field of view (10 to 100 square miles) can be processed at one time. It is nominally assumed that the initial combinations of uncertainties about the estimates for the camera parameters imply uncertainties on the ground of approximately +/- 200 feet in X and Y. The correspondence procedure works iteratively to refine the camera parameters. A typical goal is to reduce the implied uncertainties on the ground to about +/- 2 feet in X and Y.

After the image is placed into correspondence with our map data base, one or more of the visible road sections are selected for monitoring. The road center line and lane boundaries are found to an accuracy of one to two pixels in imagery with a resolution of 1 to 3 feet/pixel.

Given the precise road locations in the image, anomalous objects are detected by scanning on and along the road pavement. These anomalous objects are then identified as to type (e.g., vehicle, shadow, road surface marking, signpost, etc.).

The above tasks are supported by information about road condition and general structure from a symbolic data base. For example, if prior photographic coverage of the area being analyzed is available, the problem of anomaly classification can be simplified by determining whether a similarly shaped anomaly can be found in the same general location over some extended period of time. Additional examples of how data-base knowledge and stored models can aid in the analysis process include: using the time of day in discriminating shadows from objects of interest; utilizing the general shape and width of the road (obtained from a map) as an aid in road tracking; and providing relevant information on the anticipated size, shape, and road orientation of potential vehicles.

A central theme of this effort is to consider roads as a knowledge domain. In particular, we are addressing the question of how a priori knowledge can be directly invoked by the image-analysis modules (what type of knowledge, how should it be represented, and what are the mechanisms for its use). To achieve our goal of building a very-high-performance system, we are developing explicit models of the image structures we are dealing with and, additionally, models of the decision procedures embedded in the image-processing algorithms so that the algorithms can evaluate their own performance. Finally, we are planning an overall control structure that will be concerned with the problems of coordinating analysis across a spectrum of levels of resolution and with integrating multi-source information.

PROGRESS

Our work to date has provided the capabilities necessary to assemble an integrated Road Expert demonstration system, and we are currently planning to have such a system operational by October 1979. This system will allow a user to submit new photographs from a previously "instantiated" site for automatic analysis in which image scanning, image-to-data base correspondence, road marking, and anomaly analysis will be performed "on line".

The demonstration system will also permit both interactive instantiation of a new site and selected analysis functions (such as road tracking) on photographs for which there is no data base support.

We have previously described [2, 3] our approach to both the correspondence and road marking tasks; work continues in these two areas, both to achieve higher performance and to generalize the techniques to a wider class of domains. A more detailed description of this ongoing work will be deferred until a later time.

In the following two subsections we will describe recent progress in dealing with the problem of vehicle detection and anomaly analysis, and we will discuss our plans for on-line site instantiation.

A. Progress in Anomaly Classification

We now have a program that will analyze the anomalies detected by the correlation road tracker [3] and decide whether or not they arise from a vehicle. If an anomaly is judged to be a vehicle, then the program will provide a limited amount of classification as to vehicle type. If the anomaly is judged to be something other than a vehicle, the program provides the most likely interpretation of what it is.

The correlation road tracker has been slightly modified to produce, in addition to the road track, an image array containing the difference between the actual brightness in the original image and the brightness predicted from the road model (originally this additional output was in the form of a binary anomaly mask). The value of this "difference image" is twofold: it can be thresholded to decide what is and is not anomalous, and the image with the road profile subtracted out is useful for analyzing shadows and road discolorations.

It turns out that an understanding of shadows is crucial in making sense out of road scenes. Aerial scenes are often photographed in direct sunlight, and vehicles on the road cause anomalies that include the vehicle plus its shadow. Large objects off the road, such as signs, trees, and utility poles cast shadows that are noticed by the anomaly detector. In addition, the shadows can give valuable clues to the size and shape of the objects casting them.

We employ three basic techniques to identify shadows. A brightness model allows us to identify shadows by the absolute brightness of pixels in the difference image. A predictive model allows us to identify the portion of an anomaly most likely to be shadow when we know the position of the sun and the height of the object casting the shadow. Finally, a projective model, which tries to locate the two long parallel sides of a vehicle, can locate the dividing line between a vehicle and its shadow.

A number of "expert subroutines" examine each anomaly. The vehicle expert subroutine exploits the basically rectangular shape of vehicles when viewed from above. Anomalies that are very much the wrong size are eliminated at the outset. Projecting the average brightness and average gradient magnitude upon a baseline perpendicular to the presumed direction of vehicle travel enables finding the shadow and establishing a nominal width for the vehicle. Height can usually be estimated from the shadow, and length is inferred from the size of the total anomaly (allowing for a shadow fore or aft).

Two other anomaly experts, the tree-shadow expert and the road marking expert, provide alternate explanations for anomalies not identified as vehicles. To qualify as a tree shadow (or the shadow of some other object off the road) an anomaly must have the appropriate average brightness, a low variance in brightness, and touch the side of the road at the side nearer the sun. Road markings (usually painted arrows or speed limit numerals) are usually brighter than the road

surfaces, have low brightness variance, and are quite limited in extent.

A detailed discussion of the above material is contained in a companion paper by Gerald J. Agin [4].

9. The Road Data Base and Its Construction

This subsection describes the present state of implementation of the road data base and plans for the October 1979 demonstration involving on-line site instantiation.

The purpose of the road data base is to enable the Road Expert to accurately and reliably find known roads in new images, trace their paths, and locate anomalies that might be potential vehicles on the roads. The data base also contains information to help distinguish vehicles from permanent road features such as signs and their shadows, and painted markings on the road surface.

The current road data base contains both geometric and photometric information. The geometric part of the road data base was generated by a variety of means, depending on the level of detail and accuracy desired. The coarsest level of data representation was generated by specifying approximate world location, direction, and width of road segments, either by typing in numerical information or by tracing the road in a low resolution (USGS 7.5 minute series) map of the area. The most accurate geometric information was entered into the data base both by typing in precise numerical data and by manually tracing portions of "as built" survey plans of the road obtained from the California Department of Transportation.

Photometric information associated with a road segment is inserted into the data base by using the correlation road tracker; as images of a geographic site are interpreted by the road tracker, road photometry models are automatically entered. Spatially fixed landmarks, such as painted road surface markings, are (at present) manually specified; and a corresponding rectangular image patch is entered into the data base.

The data base is currently implemented using SAIL record structures which conveniently provide graph structures, lists, numeric arrays, etc. A general-purpose record structure I/O package communicates these structures between SAIL programs and disk files. We recognize the need, in the future, to develop a file representation that can be communicated to LISP programs.

We intend to include examples of data base construction as a part of the Road Expert demonstration and are working toward a scenario of the following type. An image of a site will be scanned and digitized at approximately one to three feet per pixel resolution; and a photo interpreter will then indicate the approximate locations of primary road segments in the image, using a track ball. The automatic road tracker program will be invoked to accurately trace the roads, generate cross-section photometry models, and detect anomalies that might be permanent surface markings. The anomaly analysis techniques described in the preceding subsection will specify which anomalies are to be included as point features in the data base. The photo interpreter will then review and edit the results.

Since a single image will not provide terrain elevation information, we are hoping to proceed as follows: After one image of a stereo pair has been analyzed as described above, the second image of the pair will be scanned and digitized. The second image will be used to determine relative elevations of road points by parallax measurements made on road surface features or nearby image areas that can be aligned by cross-correlation. Real world x,y,z will be determined from knowing the world location of a few recognizable landmarks in the images.

CONCLUDING COMMENTS

We see the military relevance of our work extending well beyond the specific road monitoring scenario presented above. In particular, a Road Expert can be applied to such problems as:

(1) Intelligence--Monitoring roads for movement of military forces

(2) Weapon guidance--Use of roads as landmarks for "map-matching" systems

(3) Targeting--Detection of vehicles for interdiction of road traffic

(4) Cartography--Compilation and updating of maps with respect to roads and other linear features (especially those concerned with transportation), such as airport runways, railroads, rivers, etc.

In accord with our generalized view of the applicability of the Road Expert and the knowledge-based, image-analysis techniques we are constructing, we are attempting to achieve a level of performance and understanding in each of the functional tasks that far exceeds that which would be required for dealing with the road-monitoring scenario alone.

REFERENCES

1. H. G. Barrow et al., "Interactive Aids for Cartography and Photo Interpretation: Progress Report, October 1977," Proceedings: Image Understanding Workshop, pp. 111-127 (October 1977).

2. M. A.Fischler et al., "Interactive Aids for Cartography and Photo Interpretation," Semiannual Technical Report, SRI Project 5300, SRI International, Menlo Park, California (October 1978).

3. L. Quam, "Road Tracking and Anomaly Detection," Proceedings: Image Understanding Workshop, pp. 51-55 (May 1978).

4. G. J. Agin, "Knowledge-Based Detection and Classification of Vehicles and Other Objects in Aerial Road Images," Proceedings: Image Understanding Workshop (in press, April 1979).

SPATIAL UNDERSTANDING

Thomas O. Binford

Artificial Intelligence Laboratory, Computer Science Department
Stanford University, Stanford, California 94305

Abstract

We describe a program of research which seeks mechanisms to solve fundamental problems in image understanding. The research is applied to a set of problems from cartography, photointerpretation and guidance. We aim to integrate solutions across this range of problems by defining a hierarchy of representations within a single system, ACRONYM. Past work has laid the foundation of representation by generalized cones, and use of this representation in identification of complex objects. We have built a system which demonstrates automated stereo registration and separation of objects from the ground surface. Stereo and motion parallax perception algorithms based on area correlation and edge matching have been developed. ACRONYM, the model-based vision system, has been designed and largely implemented. Recent work with ACRONYM has concentrated on expanding its geometric reasoning capability while concentrating on identifying aircraft at an airfield. Brooks has demonstrated ACRONYM's rule-based mechanism for segmenting images into well-formed regions. Work in stereo has been applied to navigation using passive sensing.

Introduction

This report covers the Image Understanding program at Stanford University, known as Spatial Understanding, and the Image Understanding program with Lockheed as principal contractor, concerning navigation using passively sensed images. We have chosen typical tasks from cartography, photointerpretation, and guidance as focus for our research.

1. A photointerpreter monitors an airfield. The system identifies and counts aircraft at frequent intervals to monitor air traffic.
2. An interpreter monitors a building complex for changes. The system uses stereo, a model of the complex, and identification to distinguish insignificant from significant changes. The system might not notify the interpreter about changes from snow or rain, or moving a vehicle, but notify him about building additions.
3. An interpreter monitors vehicles in staging areas. The system identifies vehicles and monitors traffic to and from the area.
4. A low-flying craft navigates with a flight path which may be changed at will, with low memory requirements from images or symbolic reference maps using passive sensing by sequences of fixes using stereo and motion parallax vision.

Reports in these proceedings describe progress with task 1 [Binford and Brooks] and task 4 [Firschein et al]. Previous reports show promising results with task 3 [Binford 1977]. We think these results sufficient to predict success soon in demonstrations of these tasks. Further, we believe that the ACRONYM system which supports the first three tasks will generalize to many other problems. To carry these demonstrations from a first success to reliable accomplishment requires better image feature description and improved ability to use shape in interpretation.

Our research has focused on ways to use shape in interpretation and ways to determine and describe surface shape from stereo or motion parallax. Traditionally, Image Understanding tasks have been framed as image matching tasks. We take a very different approach. Image Understanding tasks are spatial matching tasks. There begins to be an understanding that classic image matching encounters difficulties for some important problems. The image matching paradigm is: predict an image then match against the image. That procedure encounters severe problems; consider visible images in which surface markings or camouflage, movable objects, snow, rain, clouds, seasonal changes of vegetation all change images. While it is extremely difficult to predict images accurately, it is easy to predict parts of images which give spatial information. These predictions can be symbolic. The spatial matching paradigm is: predict spatial information in the image; infer spatial structure from the image; match reference structure with the perceived structure. The ability to infer spatial structure is a key research topic, yet one in which there is beginning to be progress. It is a medium step from image matching to spatial matching, but a step with great payoff.

ACRONYM

ACRONYM is a model-based image understanding system. It demonstrates mechanisms for interpretation of images of generic object classes in generic viewing conditions. ACRONYM is designed to be generalizable. It incorporates a powerful geometric modeling capability with a high level modeling language for natural communication with the user in terms of object models. A user gives high level descriptions of both generic and specific instances of objects. A rule-based inference system produces a symbolic summary of the predicted appearance of the objects. This geometric reasoning capability enables the system to incorporate and relate knowledge and information at different levels. The summary of predicted appearance drives a powerful syntactic matcher to find instances of the objects among features obtained by image segmentation procedures.

ACRONYM is intended to be generalizable in the following sense: Systems for different tasks should be constructed from a large core of common modules and a small set of task-specific modules. It is being tested on aircraft initially, later on vehicles, then buildings.

The system has not yet found an instance of an object in a digitized image given only a high level description of the object class and low level descriptors of the image. However, that achievement appears near. All the necessary mechanisms for such a test are operational in at least prototype form, and have been tested individually or as subsystems smaller than the total ACRONYM system. As more rules are written for the Predictor and Planner we expect to soon be able to run a complete test.

The High-level Modeler has been used to construct a large number of models. Object models are graphs whose primitive parts are generalized cones. These models are very compact. They have a natural set of levels of detail which is utilized for efficiency in identification. There is a high level language for modeling. A geometric editor beginning with interactive facilities like GEOMED, a library of primitives, and extending to rule-based reasoning about spatial relations. We are involved with mathematical analysis of generalized cone representation and analytic hidden surface algorithms.

Given models of objects, ACRONYM attempts to find instances of the objects in images. The Observability Graph will tell the matcher how to find instances. It is a symbolic summary of the expected appearance of objects in the image. It contains generic and specific predictions about shape elements and relations between them, with information about how to find them, what conclusions to draw if identified and what to conclude if they are not there. If information about viewing angles, distance and conditions is available, it can be used to produce more definite predictions. The Predictor and Planner module is a rule-based system which uses the Object Graph to produce the Observability Graph.

The program must choose features of the object which correspond to image and surface properties which segmentation programs can find. We call such features "observables". The features which we are using first include shape and two dimensional spatial relations of shapes within the image. When dealing with a stereo pair of pictures we also use three dimensional spatial relations and surface information.

Since the exact disposition of objects in the image is not known in advance, the Observability Graph can not contain an exact prediction of what will be seen. Rather, it must consist of predictions which adequately describe a range of possible appearances, generic with respect to both object class and viewpoint. They can best be thought of as supplying constraints on the way the picture can be expected to look. For instance, given that the system will be examining aerial photographs of airplanes on the ground, the predictor and planner can tell the matcher that when a candidate fuselage is found, wings should be found adjacent to it, with bilateral symmetry. There will be two possibilities for the relative angle, and once the actual angle has been found, the front and rear of the fuselage will have been distinguished. Then the finer task of locating the rear stabilizers (for positive identification of airplane type) can be carried out, confined to a small part of the image.

The best sort of observables to put into the Observability Graph are those which are invariant. This is a very strong requirement, and will often be hard to meet. Often however, there are predictions which can be made which will be true under a wide range of viewpoints. The prediction of where to look for the wings is an example - sometimes a wing may be obscured by a shadow cast by a nearby building, but most airplanes will be out in the open, with both wings visible. Thus some observables are almost invariant over a wide range of viewing conditions. We call these quasi-invariant observables. Some are quasi-invariant with respect to object class, while others are quasi-invariant with respect to viewing conditions. For example all airplanes have a long cylindrical generalized cone as the fuselage (invariant with respect to object class), and from most viewpoints (especially aerial views of airplanes on the ground) the fuselage will appear as an elongated ribbon (invariant with respect to viewing conditions). Functional observables can be viewed as quasi-invariants too, as they will work well, almost independently of the function parameters. For example, the orientations of the wings within an image are functions of the orientation of the fuselage. Conditional observables are a special case.

ACRONYM is the first vision system to incorporate a general reasoning system. It is necessary because we wish to predict the appearance of generic objects, from generic viewpoints. The Predictor and Planner has been run on a specific model of an aircraft and a generic model of an airport. It produced the complete node structure of the Observability Graphs in both cases. The detailed rule base must be completed to carry out the aircraft identification.

The Matcher has been tested on a hand coded Observability Graph, matching against a hand coded Picture Graph.

Well-Formed Ribbons

Brooks describes an algorithm for linking straight edge elements produced from a bottom up line finding stage [Brooks]. Edges are linked by a best first search algorithm. Heuristics are used both to select candidate edges for linking and to prune the search tree. A list of chains of edges is the result of this stage of processing. The choice of heuristics determines the chains of edges produced. Further heuristics prune the list of chains. Finally regions, described as ribbons, are chosen so that their boundaries are approximated by the chains of edges. Individual picture elements may appear in multiple ribbons. Further heuristics can be employed to reduce this multiplicity. It is organized so that a higher level computational procedure can easily direct this low level algorithm by supplying and altering the heuristics during processing. A selection of useful heuristics is described and then shown working on an example picture, with a hand simulated control program, resulting in effective descriptions of the image.

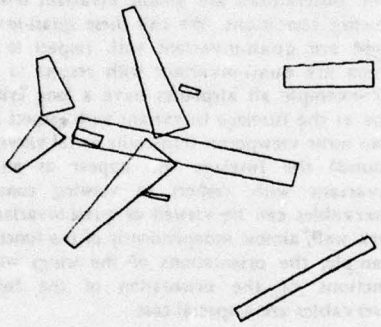


Fig. 3. An L1011.

Work is underway to transport the line finding programs of Nevatia and Babu [1978] to our laboratory. That may be used to extend goal-direction down to the line finding level. Such a capability should prove useful when the Predictor and Planner has an understanding of the process of shadow formation.

Navigation using Passive Sensing

The objective of the study is to show passive sensing techniques, particularly those based on stereo, which provide a combination of fixes on landmarks, continuous tracking of linear features where applicable, and dead reckoning. The ability to improve dead reckoning by estimating true velocity and wind drift is one benefit. A major benefit is the flexible flight planning possible. The required image base is small, particularly with linear features. Thus, a grid of check points can be accomodated.

The Stanford stereo system is being tested on images from the Night Vision Lab terrain model. The stereo system obtained automatic registration and a camera model for the two images. A terrain map will be obtained and compared with the NVL terrain map. Lockheed is evaluating the stereo system for two roles: terrain mapping of relative elevations and terrain matching; measuring Velocity/height by using the camera model solver to compute change in attitude on a non-stabilized platform. Lockheed has evaluated effects of structural flexure on absolute altitude accuracy for configurations of multiple cameras. Acceptable accuracy does seem obtainable. Of course, relative accuracy is easier to maintain.

Automatic Registration of Stereo Images

Research has demonstrated high potential for utility of passive imaging techniques for high resolution depth measurement. Passive techniques have important advantages over active ranging techniques in hostile environments. Sequences of images from a moving aircraft have been used to find the ground plane and separate objects from ground. The system should be effective with camouflaged surfaces. The accuracy attainable has been demonstrated to be 2" height error for 3" horizontal pixel size on the ground, with a 60 degree baseline. On a general purpose computer, the process requires about 15 seconds with no guidance information. That can likely be reduced at least a factor of 2. With accurate guidance information, the time required is estimated to be about 250 msec (most missions would probably be in this category). The system is self-calibrating and highly reliable.

The system includes a solution to the problem of determining the stereo camera model from information within the pair of pictures. Imagine an aircraft approaching a runway. As it moves, objects on both sides appear to move radially outward from a center, the fixed point. The fixed point is the instantaneous direction of motion. The pilot knows that the point which does not appear to move is where he will touch down, unless he changes direction. The distance between views and the apparent displacement of points allow calculation of the distance of each point from the observer and from the vehicle path. The touchdown point can be calculated from the trajectory of centers. That is precisely what is done by the camera transform solver in the system [Gennery]. It determines the transform from one view to another in a sequence of views from a moving observer.

The program first orients itself in the scene and finds a model for the transform between the two cameras. This step takes 60% of the time required for finding the ground plane. If two views are an accurately calibrated stereo pair, this operation is not necessary. If accurate guidance information is available, this operation can be speeded up enormously. The program finds a camera transform model by finding a sample of features of interest in one image and matching them with their corresponding view in the other image. The Interest Operator requires about 75 msec for a 256x256 frame. Interesting features are areas (typically 8x8) which can be localized in two dimensions without a camera transform. The operator chooses those areas with large variance along all grid directions. That is roughly equivalent to a large drop in autocorrelation along all grid directions, which means that the area can be localized closely. The correlator matches the features in the other image by a coarse to fine strategy: It has versions of the picture at resolutions of 256x256, 128x128, 64x64, 32x32, and 16x16. It first matches by a small search on a very coarse version (16x16) of

the image. It then performs a search in the next finer version of the image, in the neighborhood of the best match in the previous image. That step is repeated until the full resolution is reached. The matching process requires only 50msec for a match of a single feature no matter where it is in the image. If the camera transform is known, search is necessary only along a line in the image. In this case, search is about a factor of seven faster. If the depth of neighboring points is used as a starting point for the search, the match is another factor of seven faster. It is planned to incorporate those speedups; neither is now used. The matching has about 10% errors. It encounters fewer ambiguities than brute force matching, since not only must the feature match, but the surrounding context must also match. The procedure should not work for parts of scenes where the background of objects (context) changes drastically from one view to another. This is true only at very wide angles and close range. Aerial views are mostly planar, so failure of matching should not be a problem, nor has it been in practice. The process requires about 50k of 36 bit words now. It is possible to implement the coarse-to-fine search strategy in a raster scan and keep only a portion of each image in core. This would cut memory size by a large amount, but it has not been done.

The program automatically determines the transform between the two views. Given corresponding views of five points which are non-degenerate (i.e. no colinear and planar degeneracies) the relative transform of the two views can be found. It is not necessary to know the position of these points, only two views that correspond. The transform is determined except for a scale factor, the length of the stereo baseline. That does not affect subsequent matching of two views using the transform, and the scale factor can often be determined from known scene distances or guidance information. If the scene is nearly flat, then certain parameters are ill-determined. However, that does not affect the accuracy of measuring heights using the transform. In the present form of the camera transform solver, it sometimes encounters stability problems in degenerate cases. It may be possible to improve that. If the scene is nearly flat, then a special simplified form of the solver can be used.

The special case version has been used on some images. It is much faster than the full transform solver. Part of the job of the transform solver is to deal with mistaken matches. The procedure calculates an error matrix for each point and iterates by throwing out wild points. It calculates an error matrix from which errors in depths of point pairs are calculated. The solver uses typically 12 points and requires about 300 msec per point. It requires about 20k of memory. It requires 60% of the time for finding the ground plane. With accurate guidance information, this operation would not be necessary. However, it can be used directly to find the instantaneous direction of the vehicle. As mentioned above, as the vehicle moves, points in the image appear to move radially away from the center which is the instantaneous direction of the vehicle. Three angles relate the coordinate system of one view with the other, and two angles specify the direction of the instantaneous direction of motion.

Ground Plane

The camera transform model makes it economical to make a denser depth map. A point in one view corresponds to a ray in space which corresponds to a line in the other view. The search is limited to this line, and in addition, nearby points usually have about the same disparity as their neighbors. Thus, search is limited to a small interval on a line. A high resolution correlator has been developed which interpolates to the best match, and which calculates the precision of the match based on statistics of the area.

The system then finds a best ground plane or ground parabola to the depth points, in the least squares sense. It gives no weight to points above the ground plane; it expects many of those. It includes points below the ground plane and near the ground plane. Since points below the ground plane may be wild points, they are edited out in an iterative procedure. Of course, there may be holes. If they are small, there is no problem. If the hole is big, it becomes the ground plane. The ground plane finder requires 5 msec per point.

Edge-Based Stereo

Feature-based stereo using edges increases the accuracy with which boundaries of depth discontinuities can be found by about a factor of 25. It also provides additional information about surface markings which are not available in stereo based on area correlation. Feature-based stereo is also potentially very fast, although now area-based techniques are considerably faster. Edge-based techniques have not been developed very far, and would benefit from "smart sensor" technology. A new technique has been developed to use edge features in stereo. Edges are linked along smooth curves in 3d (in the image coordinates and in depth). The new technique is used in the object modeling and recognition modules of the system. Those edges out of the ground plane delimit bodies, if isolated.

In a stereo pair of images of an aircraft at San Francisco airport, the succession of edges matched in stereo as a function of height showed a separation in height between wing tips and and wing roots of an L1011.

Vehicle Location

Arnold demonstrated progress toward vehicle location and identification. The system registered images of a suburban parking lot and obtained the stereo camera model. It separated the vehicles from the ground and succeeded in describing the projection of a car by a rectangle of approximately the right size and orientation. The length and width of the car were accurate to about 5% by inspection. A sequence of steps are shown in figures 2 through 4 which lead to description of the car by a rectangular outline of edges above the ground.

Recognition of Complex Objects

Nevatia developed a system which took depth maps of a doll, a toy horse, a glove, a hammer and a ring using depth maps from a laser triangulation system [Nevatia 1974]. These depth maps were described in terms of generalized cones which were organized into part/whole structures. The system recognized different views of these objects with articulation of limbs and some obscuration. The system addressed important issues of representation: indexing into a subclass of similar objects in visual memory instead of matching against all stored

models; determining generalized cone descriptors from surface descriptors.

References

R.D. Arnold; *"Local Context in Matching Edges for Stereo Vision"*; Proceedings Image Understanding Workshop, Boston, May 1978.

Binford, T.O.,Brooks, R.A.;*Geometric Reasoning in ACRONYM;* these proceedings.

Binford, T.O.; *Spatial Understanding"*; in McCarthy et al, Stanford A I Lab Memo AIM-301, June 1977.

Brooks, Rodney A. [1979], *Goal-Directed Edge Linking and Ribbon Finding*, in these proceedings.

Brooks, Rodney A., Russell Greiner and Thomas O. Binford [1978a], *A Model-Based Vision System*, Proc ARPA Image Understanding Workshop, Cambridge, May., 36-44.

Brooks, Rodney A., Russell Greiner and Thomas O. Binford [1978b], *Progress Report on a Model-Based Vision System*, Proc ARPA Image Understanding Workshop, Carnegie-Mellon, Nov., 145-151.

Firschein, O., Gennery, D., Milgram, D., Pearson, J.J.; *Progress in Navigation using Passively Sensed Images;* these proceedings.

Nevatia, R; *Structured Descriptions of Complex Curved Objects for Recognition and Visual Memory* Stanford University Artificial Intelligence Laboratory, Memo AIM-250, CS-464 October 1974.

R.Nevatia and T.O.Binford; *Description and Recognition of Curved Objects;* Artificial Intelligence, Vol 8, p 77, Feb 1977.

Nevatia, Ramakant and K. Ramesh Babu [1978], *Linear Feature Extraction*, Proc ARPA Image Understanding Workshop, Carnegie-Mellon, Nov., 73-78.

Figure 2. Features of interest

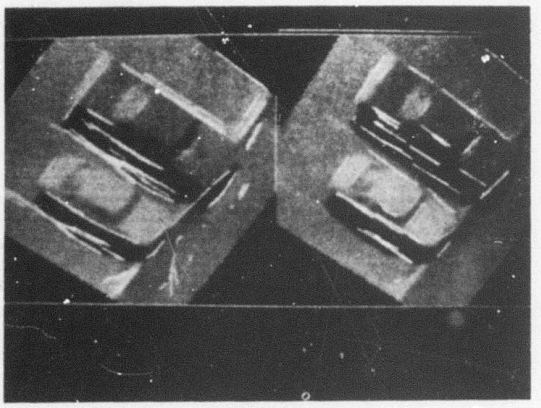


Figure 3

Linked edges near ground (left) and above ground

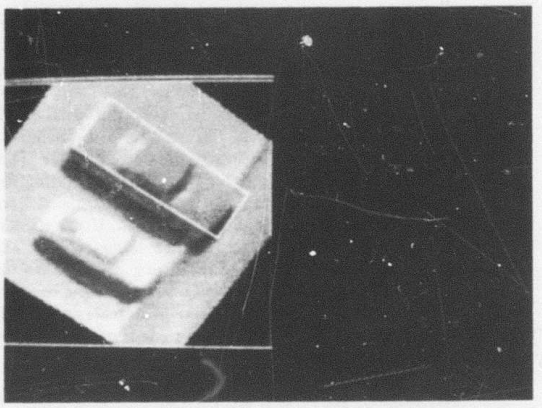


Figure 4 Rectangular parallelepiped fit

IMAGE UNDERSTANDING RESEARCH AT USC

Ramakant Nevatia and Alexander A. Sawchuk

Image Processing Institute
University of Southern California
Los Angeles, California 90007

The following is a brief summary of our recent work. The details of this work can be found in a forthcoming technical report [1].

IMAGE UNDERSTANDING PROJECTS

We have continued the development of an overall system for locating desired features in aerial images. A model is supplied to this system by a user via an interactive dialog system. Our system uses both edge and region segmentation techniques, guided by a user supplied model. Symbolic descriptions of images are matched with a symbolic model for locating desired features. Figure 1 shows an image of San Francisco area, fig. 2 shows a sketch of user supplied model, and fig. 3 shows a partial internal model. Figure 4 shows the identified regions and linear segments and desired airports are located for further analysis (shown as being bounded by rectangles). Details of this system have been described in previous progress reports [2].

Further progress has been made on our techniques for detection of roads and similar structures, bounded by locally linear and locally parallel boundary segments of opposite directions. Improved results can be obtained by bridging gaps that are caused by a single missing edge. Figure 5 shows an aerial image, linear segments detected in it, detected roads before bridging gaps, and roads after bridging gaps. Details of this technique are described in [3,4]. We have also initiated development of a program to recognize structures that are well described by such description techniques. Initially, we are using examples of airports which are usefully characterized by the arrangements of their runways and taxiways.

We have developed a new structural texture description technique that attempts to find repetitive patterns in edges of images. Periodic textures, such as in cities are easily discriminated and described. Many of the results are preliminary but promising. Details of this technique are described in a separate paper in these proceedings [5].

Statistical texture measurement techniques have been further investigated by Dr. W.K. Pratt and his associates, Dr. O.D. Faugeras, and Mr. K. Laws. On the images tested, these techniques offer high probability of correct classification. The measures are relatively simple and hardware implementation seems feasible. In another project, texture feature extraction using the singular values of a texture field as vector components has been investigated.

IMAGE PROCESSING PROJECTS

Image processing projects during the last six months are concerned with image processing system architecture, image restoration, and radar image formation. A novel architecture for performing two-dimensional convolution with a minimum amount of hardware and fewer numerical operations has been developed. The technique involves repeated sequential convolution with small generating kernels to approximate the results of convolution with large kernels.

Several projects in image restoration have been active. An algorithm for computing the condition number of a Wiener image restoration operator as a means of predicting the numerical accuracy of the restoration process has been developed. Work on image restoration for blurred images subjected to Poisson sensor noise has been continued. The restoration technique is a moving window nonlinear filter which uses maximum a posteriori (MAP) and maximum likelihood (ML) estimation criteria. The experimental results indicate some improvement over traditional linear Wiener filters, particularly in the difficult situation of signal - dependent Poisson sensor noise. Additional theoretical effort has been directed to deriving Cramer-Rao lower bounds on the theoretical estimation error. A new technique of blind deconvolution (a posteriori image restoration) using algebraic minimization of an error criterion by choice of filter weights has been developed. The experimental results indicate that the technique produces improvement with fairly small computing effort.

A final project in the image processing area concerns synthetic-aperture radar signal processing. An analysis of errors associated with data sampling in the polar domain has been

made, and a detailed treatment of synthetic aperture radar systems from an image processing context is being completed.

SMART SENSOR PROJECTS

The Hughes Research Laboratories have continued their work on the development of smart sensors for image understanding. Hughes is presently completing construction of a new CCD chip that performs the following functions:

3x3 Laplacian
5x5 median filter
5x5 programable weight convolver
7x7 bipolar convolver
26x26 edge detection convolver.

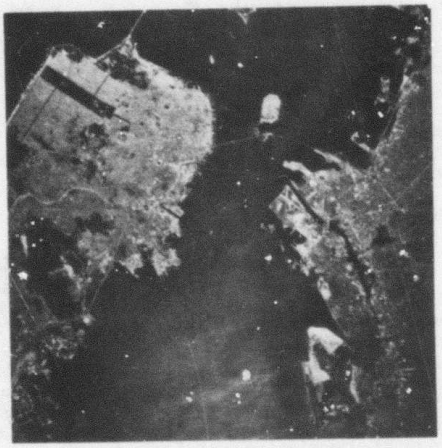


Fig.1. An aerial image of San Francisco Area.

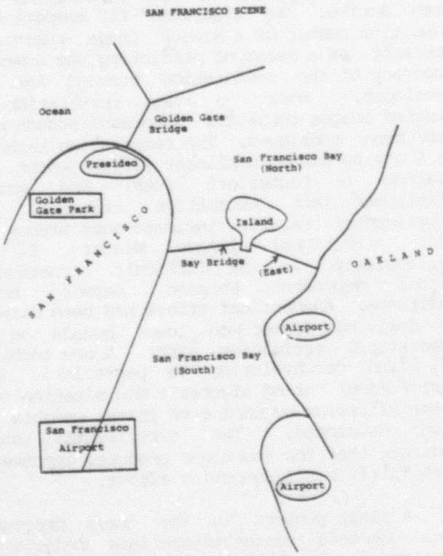


Fig.2. A schematic map.

REFERENCES

1. H.C. Andrews and W.K. Pratt (Editors), USC Semiannual Technical Progress Report, USCIPI Report 860, April 1979.

2. R. Nevatia and K. Price, "Locating Structures in Aerial Images," in USC Technical Report, USCIPI 840, October 1978.

3. R. Nevatia and K. R. Babu, "Linear Feature Extraction," in USCIPI Report 840, October 1978.

4. K.R. Babu and R. Nevatia, "Use of Linear Features for Road Detection," in USCIPI Report 860, April 1979.

5. R. Nevatia, K. Price, and F. Vilnrotter,"Describing Natural Textures," this proceedings.

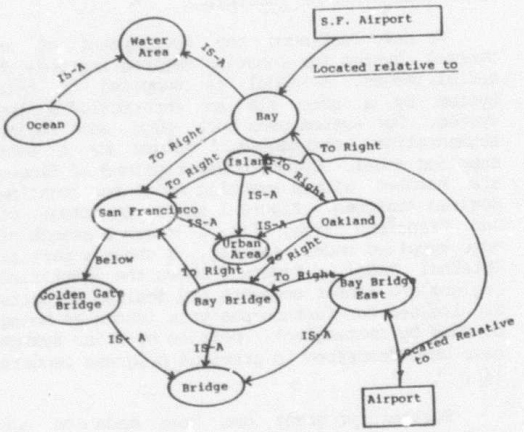


Fig.3. Internal Representation of Map.

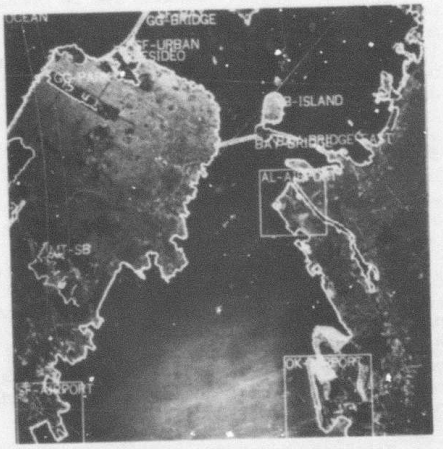
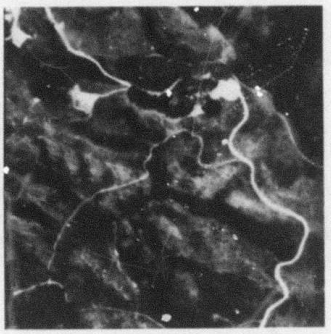


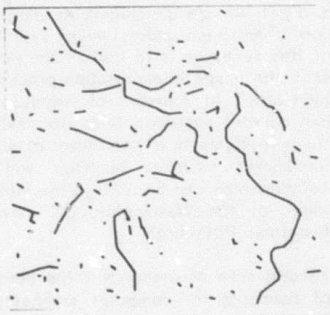
Fig.4. Recognized objects in Fig. 1.



a) An image.

b) Linear segments.

c) Detected "roads".



d) Roads after bridging one pixel gaps.

Fig. 5. Steps in processing of aerial image.

IMAGE UNDERSTANDING RESEARCH AT CMU: A Progress Report

Raj Reddy
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pa. 15213

INTRODUCTION

The primary objective of our research effort is to develop techniques and systems which will lead to successful demonstration of image understanding concepts over a wide variety of tasks, using all the available sources of knowledge. We are focusing our attention on three areas of research. First, we are developing an integrated concept demonstration of an image understanding system. The long-term goal of this research is to understand how knowledge can be used in the image interpretation process to produce systems which are 2 to 3 orders of magnitude more cost-effective than current systems. Over the next three years we expect to investigate how knowledge of maps, size and shape of landmarks such as buildings and rivers, and contextual relationships can be used in the interpretation of satellite images of the Washington, D.C. area and color scenes of downtown Pittsburgh.

The second area of research is the development and validation of concepts for computer architectures used in image understanding. The long-term objective of this research is to develop new computer architectures which will make low-cost image processing a serious possibility. We plan to evaluate the desirability of new processor designs and new instruction sets for image processing applications.

The third area is the development of intelligent interactive aids for tasks such as photo interpretation and map generation. Many of the same techniques which are useful in automatic interpretation are applicable in this area, except that in this case the human being provides the goal direction. The availability of intelligent assistants capable of examining large image data bases and retrieving desired information is expected to significantly improve human productivity in tasks such as photo interpretation and cartography.

The following is a brief summary of our work over the last six months.

KNOWLEDGE REPRESENTATION AND SEARCH

During the past six months we have concentrated on the detailed performance analysis of the ARGOS Image Understanding System (Rubin, 1978) using hand segmented data. Problems with inaccuracies in the camera model used to generate the adjacency netork and omissions in the knowledge network were identified and corrected. Further work is being done on mechanisms to resolve pointer conflict in the backtrace of the LOCUS search. These conflicts arise frequently due to the maintenance of multiple interpretations, each generated by the application of knowledge to different portions of the errorful signal data. One technique is to apply the Locus search technique to the backtrace to determine the "best" label.

We have begun to experiment with the use of a modified relaxation technique within the ARGOS Image Understanding System as an alternative to the LOCUS search. We use the same optical match, contrast, location, and adjacency knowledge producing results comparible to LOCUS but at a factor of 2 to 4 loss of speed. A report on this work is forthcoming in "An Experiment with Search Strategies for an Image Understanding System", (Smith, 1979).

IMAGE FEATURE ANALYSIS AND SEGMENTATION

A new approach to deriving three-dimensional surface orientation from image textural properties is described in "Shape from Texture: A Computational Paradigm" (Kender, 1979), in this volume. Introduced is a new representational and computational tool, the normalized textural property map, which unifes and exploits a large class of low-level image heuristics. An example of an application of the paradigm to an abstract textured image is given, and the relation of this work to existing work on shape is discussed.

We are continuing to study the effective use of knowledge in image segmentation. The KIWI segmentation program (Shafer and Kanade, in prep.) has incorporated a fast algorithm for extracting descriptions of regions resulting from a possible segmentation. By analyzing these descriptions, noise elimination can be performed without the use of global smoothing techniques. The speed of this process allows KIWI to examine, in parallel, several possible segmentations based on different image features, and to select the segmentation which results in the most viable region configuration.

The region extraction algorithm used in the KIWI program is being extended to perform other related tasks. Shafer is using this procedure to eliminate noise regions, and has found it faster than "smoothing" techniques which accomplish the same task. The procedure is less sensitive to the size of the image than smoothing, and allows more flexible definitions of "noise". The phenomenon of "degenerate histograms" has been dealt with by the same algorithm, and we are able to identify "busy" (textured) areas of an image during segmentation without preprocessing. We have solved the problem of the large data tables required by this technique, and are extending it to gather additional statistics about the regions processed.

3-D MODELING

It is a common experience for us that, given a single 2-dimensional picture of an object, we have one (or a few) definite idea(s) about its 3-D shape, in spite of the fact that a large number of possible shapes exist which produce the same picture. This fact indicates that we use some assumptions or knowledge about the objects and about the image formation.

Kanade (Kanade, 1979) has been working to identify

some of these assumptions, mostly in the geometrical aspects, by demonstrating how the theory and techniques which exploit such assumptions can provide a systematic shape-recovery method. The method consists of two parts: qualitative shape recovery and quantitative shape recovery. For the qualitative shape recovery we use a model of the Origami world (Kanade, 1978), together with edge profiles of lines taken across the line in the image in order to constrain line labels in the search of plausible interpretations.

For the quantitative shape recovery, we adopt a technique of mapping image regularities (in particular, the parallelism of lines and the skewed symmetry) into shape constraints, which is developed in (Kanade and Kender, 1979). Actual shape recovery from a single image is demonstrated for the scenes of an object such as a box and a chair. Given an image, the shape-recovery process generates a 3-D shape description of objects in terms of plane surfaces, and the description is supplied to a display program which can synthesize images of the same object as we would see it from other view directions.

INTERACTIVE AIDS

We are continuing with the integration of map knowledge of the Washington, D.C. area into our system. The map knowledge consisted of a terrain (elevation) database and cultural features such as rivers, major buildings, forests and roads. We plan to apply this knowledge in a system which will match satellite and aerial photographs to the terrain model and extract information from the images using the cultural feature data.

ARCHITECTURES FOR IMAGE PROCESSING

SPARC, the high speed processor being jointly designed by Control Data and CMU, has completed the design phase and begun layout and fabrication. Currently the four major component boards which contain the bulk of the custom LSI circuitry are being routed and fabricated. We expect that the processor will be delivered to CMU in the fall of 1979. Current gate level simulations indicate that instruction speeds in the order of 20ns can be expected.

Researchers at CMU have already designed several prototype NMOS LSI circuits utilizing graphics software running under the UNIX operating system. Work is underway to complete a design laboratory which will allow top down design of VLSI circuits, as well as provide post-fabrication packaging and testing facilities. The laboratory is intended to allow computer scientists with a minimal understanding of solid-state physics and IC design to rapidly produce working circuits. A number of special purpose chips are expected to be designed to implement common image understanding algorithms, such as edge detectors and smoothing operators.

Our collaboration with Texas Instruments (Eversole et. al., 1978) to jointly design and develop an all-digital programmable VLSI chip set for several low level vision operations has begun to result in breadboard designs for several important operators: a programmable sum of products operator and a 5x5 median operator.

CONCLUSION

While the primary emphasis continues to be in effective use of knowledge in the image interpretation process, the research at CMU is tempered by the realization that we must also pay adequate attention to other relevant aspects such as computer architecture, software design, image databases, performance analysis and perceptual psychology. We continue to have modest efforts in each of these areas.

REFERENCES

Eversole, W. L., et. al. (1978). "Investigation of VLSI Technologies for Image Processing", Proceedings of the ARPA Image Understanding Workshop, Science Applications Inc., Nov., 1978

Kanade, T. (1978). "A Theory of Origami World" CMU Technical Report, Department of Computer Science. September, 1978.

Kanade, T. (1979). "Recovery of 3-Dimensional Shapes of an Object from a Single View" CMU Technical Report, Department of Computer Science. (in preparation).

Kanade, T. and Kender, J. R. (1979). "Skewed Symmetry: Mapping Image Regularities into Shape," Technical Report, Computer Science Dept., Carnegie-Mellon Univ. (in preparation).

Kender, J. R. (1978). "Shape from Texture: A Brief Overview and a New Aggregation Transform", Proceedings of the ARPA Image Understanding Workshop, Science Applications Inc., Nov., 1978.

Kender, J. R. (1979). "Shape from Texture," Ph.D. Thesis, Computer Science Dept., Carnegie-Mellon Univ., 1979 (in preparation).

Rubin, S. (1978). "The ARGOS Image Understanding System", Ph.D thesis, Department of Computer Science, Carnegie-Mellon University.

Smith, D. (1979). "An Experiment with Search Strategies for an Image Understanding System", (in preparation).

Shafer, S. and Kanade, T. (1979 in preparation). "KIWI: A Flexible System for Region Segmentation", CMU Technical Report, Department of Computer Science.

IMAGE UNDERSTANDING PROJECT STATUS REPORT

Azriel Rosenfeld
Principal Investigator

Computer Vision Laboratory
Computer Science Center
University of Maryland, College Park, MD 20742

ABSTRACT

Current activities on this project are reviewed under the following headings:

1. Image modelling and preprocessing
2. Edge detection and linking
3. Segmentation and texture analysis
4. Pattern, shape, and structure matching
5. Hierarachical region representation

INTRODUCTION

This project is concerned with the study of advanced techniques for the analysis of reconnaissance imagery. It is being conducted under Contract DAAG-53-76C-0138 (DARPA Order 3206), monitored by the U.S. Army Night Vision Laboratory, Ft. Belvoir, VA (Dr. George Jones). The Westinghouse Systems Development Division, as a subcontractor, is investigating hardware implementation of the techniques being developed by Maryland, particularly in the area of relaxation; their efforts are reviewed in a separate paper in these Proceedings.

The preceding phase of this project [1] was concerned with tactical target detection on FLIR imagery. The current phase deals with a wider variety of imagery, and includes work on image modelling and preprocessing; edge detection and linking; segmentation and texture analysis; and matching of patterns, shapes, and relational structures. Particular emphasis is being placed on the use of convergent evidence and cooperative computation ("relaxation") in segmentation and matching. A recently initiated effort deals with hierarchical region representations ("quadtrees") and their uses in image analysis.

Work done during the first six months of the current phase was summarized in [2-3], and will be mentioned here only briefly, for background purposes. This status report deals primarily with the work done during the past six months. One topic, involving the use of relaxation in segmentation, has been singled out for more detailed discussion in a separate paper in these Proceedings.

IMAGE MODELLING AND PREPROCESSING

Mosaic models for images

Under an AFOSR grant [4], extensive work has been done on a class of "mosaic image models" based on random geometric processes. In cell structure models, the given planar region is tessellated into cells, and "colors" (e.g., gray levels) are independently assigned to these cells according to specified probabilities. Examples of such models are the Poisson line model, in which the plane is tessellated by random lines; the occupancy model, in which random points in the plane define "Dirichlet cells" consisting of the parts of the plane that are closer to each point than to any of the others; and the Delaunay model, in which the tessellation is obtained by joining all pairs of the random points whose Dirichlet cells are adjacent. In coverage (or "bombing") models, randomly oriented figures are placed at random points in the plane, and colors are assigned to the figures and background according to specified probabilities. An introduction to such models can be found in [5].

Images generated by mosaic models consist of connected components each having a constant gray level. Analytical results have been obtained on the expected values of various geometrical properties of these components, including the expected area and width (=length of intercept by a random line) of a component, the expected number of components, and the expected total perimeter in the mosaic. Results have also been obtained on the joint gray level probability as a function of separation in these mosaics, from this, the autocorrelation, edge density, and variogram (=expected squared gray level difference) can be derived. The details can be found in [4].

The patterns generated by mosaic models are much simpler than real images; more realistically, the cells should be blurred and noisy. However, if we compare the predicted values of various properties for the models with those for suitable real images, we find that the predictions obtained from the "plausible" model are much better fits to the real data than those obtained from other models. For example, consider the picture of marble (Brodatz, Plate 62) shown in Figure 1, which resembles the patterns generated by a Poisson line model. Table 1 shows the observed values of

total perimeter and of expected component width for this picture thresholded at 25 (on a 0-63 grayscale); these values are not very sensitive to the choice of threshold. Predicted values based on the Poisson line, occupancy, and Delaunay models are also shown. We see that the Possion line predictions are much better fits to the observed values. The details of these experiments can be found in [6].

In principle, one could use the predicted total perimeter values to predict the expected edge strength for an image, by averaging the expected edge strength in the interiors of cells (as obtained from a model for the gray level population in a cell) with that on intercell borders (as obtained from the mosaic model itself). However, such predictions would not be very accurate for several reasons: (1) the model assumes that the transitions between cells are sharp; (2) the amount of border (i.e., the total perimeter) cannot be predicted very accurately. (For a more detailed discussion of these problems, see [6]). Further work on the application of these models to real images is needed.

Preprocessing

Many image processing and segmentation techniques assume that ideal images are approximately piecewise constant, i.e., are composed of relatively "flat" regions separated by relatively steplike edges. (Note that the ideal images generated by mosaic models do have these properties.) This assumption provides a basis for conventional methods of image segmentation by analysis of the gray level histogram or other pixel feature space (the regions should give rise to sharp histogram peaks or compact feature value clusters) and for edge detection (the region borders should give rise to stronger edge values than the interiors).

In practice, images are noisy, and it is not obvious how to approximate them by piecewise constant functions. Noise can be reduced by local averaging, but this blurs the edges between the regions, which is also undesirable. A variety of nonlinear noise cleaning schemes have been devised that smooth noise without blurring edges. In [7], a number of these schemes are compared. Their definitions are summarized below; in each method, a new gray level P' for point P is computed as a function of the gray levels in its 3-by-3 neighborhood N(P):

a. Mode filtering: P' is the most frequently occurring gray level in N(P).
b. Median filtering: P' is the median gray level in N(P).
c. E^k: P' is obtained by averaging P with the k points of N(P) that are closest to it in gray level.
d. Gradient smoothing: P' is the average of those points of N(P) that have lower gradient values than P.
e1. Selective averaging 1: P' is the average of N(P) provided P differs from at least 6 of its neighbors by at least t.
e2. Selective averaging 2: P' is the average of N(P) provided the edge strength at P is less than t; otherwise, P' is the average of the two neighbors in the direction along the edge.
e3. Selective averaging 3: Analogous, but using four directional edge masks, rather than differences in two perpendicular directions, to determine edge strength and direction.
f. Maximum homogeneity smoothing: Five 4x4 neighborhoods surrounding P are used; P' is the average of that neighborhood which is most homogeneous.
g. Neighbor weighting (1,2): P's is a weighted average of N(P). (The definitions of the weights are somewhat complicated, and will not be reproduced here.)
h. Weighted averaging: P' is a weighted average of P and the mean of N(P), where the weight given to P depends on how high the local image variance is relative to the overall image variance. (This method was not iterated.)

For the detailed definitions of the methods, see [7]. A simple Kalman filtering scheme was also applied to the same images. The best methods were median filtering, gradient smoothing, and the first neighbor-weighting method. Evaluations were based on subjective judgment [7], on the improvement in the image's histogram (i.e., emergence or clearer separation of peaks), and on mean squared error [8].

The best few methods (the three just mentioned, and also the E^5 method) were also applied to a color image [8]. Separate noise cleaning on the individual R, G, B color components was found to be somewhat more effective than "vector" noise cleaning in the three-dimensional color space; but when a different color coordinate system (U,V,W) was used, the reverse was true. Details of the results and their evaluation can be found in [8].

A number of new image smoothing techniques have also been tested [9]; several of these give results comparable in quality to those obtained using the best methods tested in [7]. One approach makes use of half-neighborhoods; it chooses three consecutive neighbors of P whose average gray level differs maximally from that of the other five, and averages P with those five. Another approach uses a weighted average of the neighbors in which a neighbor's weight depends on how different it is from P; however, this scheme does not smooth very effectively. Still another idea is to average P with all of N(P) if it differs from the mean of N(P) by less than the standard deviation, and to average P with only its four most similar neighbors (i.e., E^4) otherwise. The details of these methods, and examples of results, can be found in [9].

With any point P of an image I we can associate a gray level probability, $\pi(P)$, estimated from the image's histogram. If we regard $\pi(P)$ as a (rescaled) gray level, the resulting image may be

called the probability transform of I. (Other types of probability transforms can be defined based on the probabilities of local properties other than gray level, or on joint probabilities; the details will not be given here. An example using joint probabilities is the "texture transform" of Haralick.) Noise cleaning, thresholding, edge detection, and other standard processing techniques can be applied to a probability transform rather than to the original image. Experiments along these lines are in progress (e.g., [9] gives some interesting noise cleaning results) and will be desribed in a forthcoming technical report.

EDGE DETECTION AND LINKING

The Roberts and Hueckel edge detectors

The classical approach to edge detection involves convolving a set of masks, representing steps in various orientations, with the image, and defining the edge strength at a point to be the max of the convolved values. The simplest scheme of this type, due to Roberts, uses the masks $\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}$ and $\begin{smallmatrix}0&1\\-1&0\end{smallmatrix}$, corresponding to the $\pm$ 45° directions. Thus in the 2-by-2 image neighborhood $\begin{smallmatrix}A&B\\C&D\end{smallmatrix}$, the Roberts edge strength is max(|A-D|,|B-C|).

A more sophisticated edge detection technique, due to Hueckel, finds a best-fitting step edge to a given image neighborhood, and takes the edge strength to be the height of this step. Hueckel used a relatively large neighborhood, and determined the best fit by expanding the step edge and the neighborhood in terms of a set of nine basis functions. Various authors (e.g., Nevatia, O'Gorman, Mero and Vassy, Hummel) have investigated simplifications of Hueckel's approach.

The simplest possible version of Hueckel's method uses a 2-by-2 neighborhood and three basis functions, namely $\begin{smallmatrix}1&1\\1&1\end{smallmatrix}$, $\begin{smallmatrix}1&1\\-1&-1\end{smallmatrix}$, and $\begin{smallmatrix}-1&1\\-1&1\end{smallmatrix}$. When the best-fitting step edge is determined using this basis, it turns out that the magnitude of this step is precisely the Roberts edge strength. The details of the proof can be found in [10].

Straight edge enhancement

There are a number of standard methods of detecting straight edges (or lines) in an image. One of the most commonly used of these is the Hough transform, in which, for each edge point P, we estimate the slope and distance from the origin of the straight line through P. In this way, the edge points are mapped into points in (slope, distance) space. Evidently, sets of collinear edge points map into approximately the same point in this "Hough space"; thus we can detect straight edges, whether continuous or broken, by looking for high concentrations of points in Hough space.

A problem that arises in the use of Hough transforms is the uncertainty in estimating the slopes of the edge points. When a standard edge detector, based on 3x3 neighborhoods, is used, these slopes are quite uncertain. Moreover, this leads to an even greater uncertainty in estimating distance if the given edge point is far away from the point where the line through it comes closest to the origin.

The usefulness of the Hough transform can be improved by increasing the accuracy of the edge slope estimates. This can be done by iteratively reestimating the slope and magnitude of edge responses at each point based on the values of the current estimates at nearby points.

The details of such an edge reinforcement process are described in [11]. Figure 2 shows an example involving an aerial photograph of an airport.

Strip detection

Iterative reinforcement can also be used to enhance parallel-sided strips, i.e., pairs of "antiparallel" edges. An example, involving vertical edges only, is given in [12]. We assume that the vertical edge strength is a signed quantity, with positive values corresponding to (low,high) transitions, and negative values to (high,low). A cross-shaped neighborhood of each edge point P is examined. The edge strengths along the vertical arm of the cross provide increments to the edge strength at P. Thus edges along this arm having the same sense as that at P reinforce P, while those having the opposite sense weaken P. At the same time, the edge strengths along the horizontal arm provide decrements to the strength at P; thus edges along this arm having the same sense as P weaken P, while those having the opposite sense reinforce P. As Figure 3 illustrates, a few iterations of this process serve to greatly enhance vertical-sided strips while weakening noise edge responses. The details of the process, and of variations on it, are described in [12].

SEGMENTATION AND TEXTURE ANALYSIS

Segmentation by relaxation

Extensive work has been done at Maryland and elsewhere on the use of iterative probability adjustment schemes ("relaxation") for classification in the presence of constraints. In particular, this approach has been applied to most of the standard methods of image segmentation by pixel classification. A package of programs has been developed to facilitate experimentation with relaxation methods at the pixel level; it will be described in a forthcoming technical report.

A recent application of relaxation to segmentation deals with thresholding, i.e., with the classification of the pixels into "light" and "dark" classes. Initially, estimated probabilities of membership in these classes are computed for each pixel on the basis of its gray level (i.e., proportional to the distances of its gray level from the ends of the gray level range). These probabilities are then iteratively adjusted based on the neigh-

boring probabilities, light reinforcing light and dark dark. (This process is analogous to, but not the same as, image smoothing by iterated local averaging. Rather than smoothing the picture, it tends to drive light gray levels toward white and dark ones toward black.) An example of the operation of this process is shown in Figure 4; it will be described in more detail in a forthcoming technical report. Other versions of this approach, in which the classes need not be "light" and "dark," but can be defined appropriately for the given picture, are also under study.

Relaxation has also been applied to edge (or curve) enhancement. Here edge probabilities are initially assigned to each pixel based on the relative responses of edge masks in various orientations, and a "no edge" probability is also assigned based on their absolute responses. These probabilities are then adjusted iteratively based on the neighboring probabilities. In particular, edge reinforces edge to the extend that they smoothly continue one another; no edge reinforces no edge; edge reinforces no edge alongside it, and competes with no edge at its end.

The light/dark and edge/no edge relaxation processes can also be combined; e.g., edge reinforces "light" alongside it on its light side, and reinforces "dark" alongside it on its dark side, and so on. This combination gives better results than either of the two processes operating individually. Experiments using this approach are described in a separate paper in these Proceedings.

Using the combined relaxation process for image segmentation is analogous to using "convergent evidence," involving both gray level and edge value, in segmentation. In the "Superslice" convergent-evidence scheme, developed under an earlier phase of this project, thresholding and edge detection are applied separately; when both give consistent results, we say that an object has been detected. In the relaxation approach, gray level and edge evidence interact from the beginning; if they are consistent, they will mutually reinforce, making object detection easy. This has the potential advantage that it does not involve information-destructive steps (thresholding, edge maximum selection), as Superslice did.

Texture analysis: gray level cooccurrence based on edge maxima

One of the most effective current approaches to texture classification employs properties derived from second-order gray level probability densities. For any displacement $\underline{\delta}$, let $p_{\delta}(z,w)$ be the probability that a pair of points at separation $\underline{\delta}$ have gray levels z and w. The matrix whose (i,j) entry is $P_{\delta}(i,j)$, where i,j run through the possible gray levels, is called a gray level cooccurrence matrix. Statistics computed from such matrices have proved to be very useful as textural properties.

Davis has recently suggested a generalized approach to cooccurrence analysis in which joint probabilities are estimated only for pairs of "special" points, e.g., local feature points. Moreover, instead of joint gray level probabilities, we can use joint probabilities of other properties associated with a feature point. The point pairs need not be at a fixed separation; we can define $\underline{\delta}$ at each feature point based on the properties of that point. For example, we can use edge strength maxima as feature points; we can take $\underline{\delta}$ to be in the direction along or across the edge at the given point; and we can estimate cooccurrence probabilities for pairs of edge slopes obtained in this way. This yields what we may call a slope cooccurrence matrix. The slope pair statistics should provide useful information about the sizes and shapes of the texture elements defined by the edge maxima.

Alternatively [13], we can define texture properties based on the gray level cooccurrences for pairs of points one or both of which are edge strength maxima. Specifically, at each edge maximum P we look for another one (within some bounded distance), say Q, in the direction along or across the edge, and record the gray levels at P and Q; the numbers of such pairs having given gray levels provide estimates of the desired joint probabilities. Alternatively, we simply move a given distance δ from P in the direction along or across the edge, and pair the gray level found there with that at P. The resulting gray level statistics will also depend on the texture element sizes and shapes.

Pilot experiments [13] indicate that texture properties derived in these ways are not always as effective for texture discrimination as the classical gray level cooccurrence statistics. For example, Figures 5a and 6a show three samples each of three terrain types and four textures from Brodatz' album, and Figures 5b and 6b show the corresponding edge maxima. Note that two of the terrain types are hard to discriminate based on the pattern of edge maxima alone. Not surprisingly, properties derived from the maxima are usually not effective in separating these classes, as seen in Table 2. The maxima-based properties are more effective in discriminating the Brodatz textures, as Table 3 shows.

PATTERN, SHAPE, AND STRUCTURE MATCHING

Feature pattern matching

Correlation-based methods of matching two images of the same scene are quite sensitive to relative distortion. One way to overcome this is to extract a set of feature points from each image, and match the resulting feature patterns. The computational cost of feature point matching grows only with the number of feature points, not with the picture size. Good match peaks can be obtained even if many of the points detected in one image are absent from the other one and vice versa, and the matching process can be made insensitive to appreciable amounts of distortion. Experiments using this approach are described in

[14-15]; some of these were summarized at the preceding workshop [16]. Fuzzy relaxation methods can also be used as an aid in matching feature patterns, as described in [16,17].

Relational structure matching

Relaxation methods are also quite effective for matching relational structures, e.g., for finding matches between a "model graph" and subgraphs of a given "scene graph". Experiments using this approach are described in [18-19], and were summarized at the preceding workshop [16]. For symbolically labelled graphs, where exact matching is required, "discrete relaxation" is appropriate [18]; for numerically labelled graphs, where the matching is quantitative, one can use fuzzy relaxation [19].

Shape matching

A method of applying relaxation to ambiguously segmented one-dimensional patterns has been developed under an NSF grant [20], and has been applied to the disambiguation of handwritten words [21]. (This application makes use of a new probabilistic relaxation formula [22] which can be easily extended to allow interactions between triples of nodes rather than pairs; this allows trigram frequencies to be used in defining node label compatibilities.) Preliminary experiments have been conducted on the application of this approach to segmentation and labeling of the boundary of a shape.

An example, involving an airplane shape, is shown in Figure 7. (A report on this work is in preparation; we give here only a sketchy description.) In this example, the boundary arc between any pair of negative (=concave) curvature peaks, not necessarily consecutive, was taken to be a segment provided its length was between 10% and 50% of the perimeter. Initial probabilities for the labels "nose", "tail", "Lwing", and "Rwing" were assigned to each segment based on some simple shape features. Table 4 lists the 28 segments and the initial probabilities.

In the relaxation process, a label was dropped if it was not part of a compatible consecutive triple. Table 5 shows the numbers of segments (having at least one surviving label) and labels remaining after each iteration. After three iterations, the process had stabilized, and each node had only one label, as shown in Table 6. All cycles of length 4 were found in the resulting graph, and their compound probabilities were computed, as shown in Table 7. There were only six such cycles, and the three with high probabilities correspond to approximately the same segmentation, which is the plausible one.

HIERARCHICAL REGION REPRESENTATION

Region representation plays a key role in image and scene analysis, computer cartography, and computer graphics. There are a variety of approaches to representing regions, based on their boundaries or their "skeletons"; some of these are reviewed in the following paragraphs. Recently, a tree representation has been proposed which offers a number of advantages; it is also described below. Since each type of representation has its own advantages, it becomes desirable to develop efficient methods of converting from one representation to another.

The boundary of a simply-connected region is specified, relative to a given starting point, by a sequence of "unit" vectors in the principal directions. Such "chain codes" provide a very compact region representation, and make it easy to detect features of the region boundary, such as sharp turns ("corners") or concavities. On the other hand, it is harder to determine properties such as elongatedness from a chain code, and it is also difficult to perform operations such as union and intersection on regions represented by chain codes.

Another class of region representations involves various types of maximal "blocks" that are contained in a given region. For example, we can represent a region R as a linked list of the runs (of pixels) in which R meets the successive rows of the array. Here each "block" is a 1-by-m rectangle, where m is the run length; the runs are the largest such blocks that R contains, and R is determined by specifying the initial points (or centers) and lengths of the run. Alternatively, we can represent R by the set of maximal square blocks (or blocks of any other desired shape) that it contains; here R is determined by specifying the centers and radii of these blocks. This representation is called the medial axis transformation, or MAT. It is somewhat less compact than a chain code, but it has advantages with respect to performing union and intersection operations or detecting properties such as elongatedness (in terms of the smallness of the radii relative to the number of centers).

There has been recent interest in an approach to region representation based on successive subdivision of the array into quadrants. If the region does not cover the entire array, we subdivide the array, and repeat this process for each quadrant, each subquadrant,... as long as necessary, until we obtain blocks (possibly single pixels) that are entirely contained in the region or entirely disjoint from it. This process can be represented by a tree of degree 4 (for brevity: a quadtree) in which the entire array is the root node, the four sons of a node are its quadrants, and the leaf nodes correspond to those blocks for which no further subdivision is necessary. Since the array was assumed to be 2^n-by-2^n, the tree height is at most n. This method of region representation was proposed by Klinger; it has also been used for image representation. It is relatively compact, and is also well suited to operations such as union and intersection, and to detecting various region properties. A recent Ph.D. thesis by Hunter in the domain of computer graphics develops a variety of algorithms for the manipulation of quadtree region representations.

Those algorithms, however, allow a node to store the list of coordinate points that describe the polygon from which the quadtree was constructed.

Since the quadtree and border representations both have computational advantages, it is of interest to develop methods of converting from one representation to the other. Algorithms for both of these tasks have been developed, and are described in two reports [23,24]. Further work is in progress on the use of quadtree representations for such purposes as connected component labeling and shape matching.

REFERENCES

1. Algorithms and Hardware Technology for Image Recognition, Final Report, March 1978.
2. Image Understanding Using Overlays, Semi-annual Report, 1 April-30 September 1978.
3. Project status report, "Image Understanding Using Overlays", Proceedings, Image Understanding Workshop, November 1978, 20-27.
4. N. Ahuja, Mosaic Models for Image Analysis and Synthesis, Ph.D. Dissertation, University of Maryland, March 1979.
5. B. Schachter and N. Ahuja, Random pattern generation processes, Computer Graphics Image Processing, in press.
6. T. Dubitzki, N. Ahuja, and A. Rosenfeld, Predicted properties of mosaic images, TR-746, Computer Science Center, University of Maryland, College Park, MD, March 1979.
7. J. P. Davenport, A comparison of noise cleaning techniques, TR-689, Computer Science Center, University of Maryland, College Park, MD, September 1978.
8. J. P. Davenport and A. Rosenfeld, A comparison of noise cleaning techniqeus for color images, TR-748, Computer Science Center, University of Maryland, College Park, MD, March 1979.
9. A. Scher, F. R. D. Velasco, and A. Rosenfeld, Some new image smoothing techniques, TR-733, Computer Science Center, University of Maryland, College Park, MD, February 1979.
10. A. Rosenfeld, The simplest "Hueckel" edge detector is a Roberts operator, TR-747, Computer Science Center, University of Maryland, College Park, MD, March 1979.
11. S. Peleg and A. Rosenfeld, Straight edge enhancement and mapping, TR-694, Computer Science Center, University of Maryland, College Park, MD, September 1978.
12. A. Danker and A. Rosenfeld, Strip detection using relaxation, TR-725, Computer Science Center, University of Maryland, College Park, MD, January 1979.
13. C. R. Dyer, T. Hong, and A. Rosenfeld, Texture classification using gray level cooccurrence based on edge maxima, TR-731, Computer Science Center, University of Maryland, College Park, MD, March 1979.
14. D. J. Kahl, A. Rosenfeld, and A. Danker, Some experiments in point pattern matching, TR-690, Computer Science Center, University of Maryland, College Park, MD, September 1978.
15. D. J. Kahl, Sketch matching, TR-716, Computer Science Center, University of Maryland, College Park, MD, November 1978.
16. A. Rosenfeld, Some experiments in matching using relaxation, Proceedings, Image Understanding Workshop, November 1978, 110-119.
17. S. Ranade and A. Rosenfeld, Point pattern matching by relaxation, TR-702, Computer Science Center, University of Maryland, College Park, MD, October 1978.
18. L. Kitchen, Discrete relaxation for matching relational structures, TR-665, Computer Science Center, University of Maryland, College Park, MD, June 1978.
19. L. Kitchen, Relaxation applied to matching quantitative relational structures, TR-707, Computer Science Center, University of Maryland, College Park, MD, October 1978.
20. F. R. D. Velasco and A. Rosenfeld, The application of relaxation to waveforms with ambiguous segmentations, TR-683, Computer Science Center, University of Maryland, College Park, MD, August 1978.
21. S. Peleg, Ambiguity reduction in handwriting with ambiguous segmentation and uncertain interpretation, TR-724, Computer Science Center, University of Maryland, College Park, MD, January 1979.
22. S. Peleg, A new probabilistic relaxation scheme, TR-711, Computer Science Center, University of Maryland, College Park, MD, November 1978.
23. C. R. Dyer, A. Rosenfeld, and H. Samet, Region representations: boundary codes from quadtrees, TR-732, Computer Science Center, University of Maryland, College Park, MD, February 1979.
24. H. Samet, Region representation: quadtrees from boundary codes, TR-741, Computer Science Center, University of Maryland, College Park, MD, March 1979.

	Perimeter	Width
Observed	2013	20.95
Poisson line	2695	19.07
Occupancy	8250	11.86
Delaunay	8055	12.19

Table 1. Observed and predicted values of total perimeter and expected component width for Figure 1b.

Cooccurrence Type	Method	Feature ASM	CON	ENT	COR
Gray level	$\delta = (0,2)$	B/A,C	A/B/C	B/A,C	A/B/C
Edge maxima	Along edge	poor	poor	poor	A/B
	Across edge	poor	poor	poor	A/B
Gray level at and near edge maxima	Most similar neighbor along edge	poor	poor	poor	poor
	Most similar neighbor across edge	poor	B/C	poor	poor
	Least similar neighbor across edge	poor	B/C	poor	poor
	Pair of neighbors across edge	poor	poor	poor	C/A,B
	All neighbors along edge	poor	poor	poor	A/B
	All neighbors across edge	poor	poor	poor	poor

Table 2. Class separabilities for the terrain samples using various types of cooccurrence-based properties (/ means "separated").

Cooccurrence Type	Method	Feature ASM	CON	ENT	COR
Gray level	$\delta = (0,2)$	W/G,R,S	G/W/R,S	G/W/R,S	G/W/R,S
Edge maxima	Along edge	W/R,S	R/S/G,W	W/G,R,S	G/R/S,W
	Across edge	poor	R/W/G,S	W/G,S	R/G,S,W
Gray level at and near edge maxima	Most similar neighbor along edge	R/G	poor	G/R/S/W	R/G,S
	Most similar neighbor across edge	R/G	poor	G/R/S/W	R/S,W
	Least similar neighbor across edge	R/G	G,S/R,W	G/R/S/W	G/R/W
	Pair of neighbors across edge	G,S/R,W	W/G,S	G/R/S/W	R/G,S,W
	All neighbors along edge	G/R/W	W/G,S	G/R/S/W	G,S/R,W
	All neighbors across edge	G/R	G/R,W	G/R/S/W	G/R,W

Table 3. Same as Table 2 for the Brodatz samples.

Segment Number	Range of Segment	Probabilities			
		NOSE	RWING	TAIL	LWING
1	4 - 8	0.60	0.24		0.16
2	4 - 10	0.52			0.48
3	4 - 24		1.00		
4	4 - 26				1.00
5	5 - 8	0.60	0.20		0.20
6	5 - 10	0.48			0.52
7	5 - 24		1.00		
8	5 - 26				1.00
9	6 - 8	0.60	0.18		0.22
10	6 - 10	0.46			0.54
11	6 - 24		1.00		
12	6 - 26				1.00
13	8 - 24	0.30	0.70		
14	8 - 26				1.00
15	10 - 24	0.60	0.38		0.02
16	24 - 30		1.00		
17	24 - 32		0.18	0.60	0.22
18	26 - 28	0.60	0.38		0.02
19	26 - 30	0.04			0.96
20	26 - 32				1.00
21	28 - 30	0.60	0.18		0.22
22	30 - 8		1.00		
23	30 - 10		1.00		
24	32 - 4	0.46			0.54
25	32 - 5	0.42			0.58
26	32 - 6	0.40			0.60
27	32 - 18				1.00
28	32 - 10				1.00

Table 4. Segments and label probabilities for Figure 7.

Iteration Number	Number of Nodes	Total Number of Labels at all Nodes
0	28	50
1	20	25
2	17	17
3	12	12

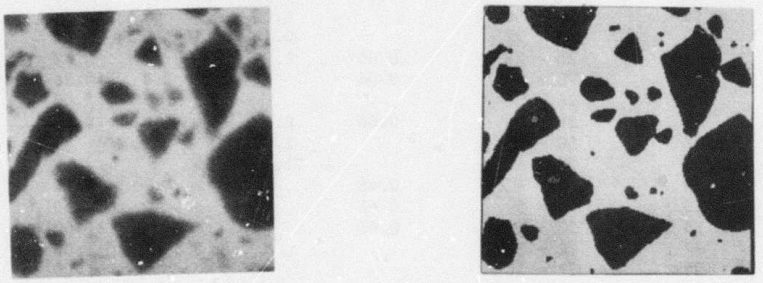
Table 5. Numbers of nodes and labels remaining at each iteration.

Segment Number	Range of Segment	Final Label
1	4 - 8	NOSE
2	4 - 10	NOSE
5	5 - 8	NOSE
6	5 - 10	NOSE
9	6 - 8	NOSE
10	6 - 10	NOSE
13	8 - 24	RWING
15	10 - 24	RWING
17	24 - 32	TAIL
24	32 - 4	LWING
25	32 - 5	LWING
26	32 - 6	LWING

Table 6. Nodes surviving at the third iteration, with their unique labels.

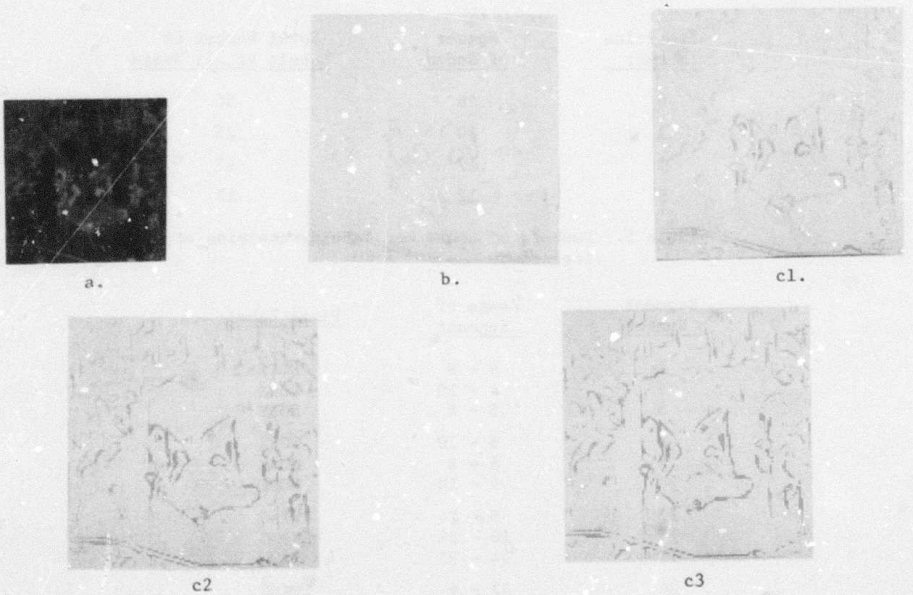
Cycle Number	NOSE Node	RWING Node	TAIL Node	LWING Node	Merit
1	1	13	17	24	13.6
2	2	15	17	24	6.4
3	5	13	17	25	14.6
4	6	15	17	25	6.3
5	9	13	17	26	15.1
6	10	15	17	26	6.3

Table 7. 4-cycles at the third iteration, with their (original) compound probabilities.



a. b.

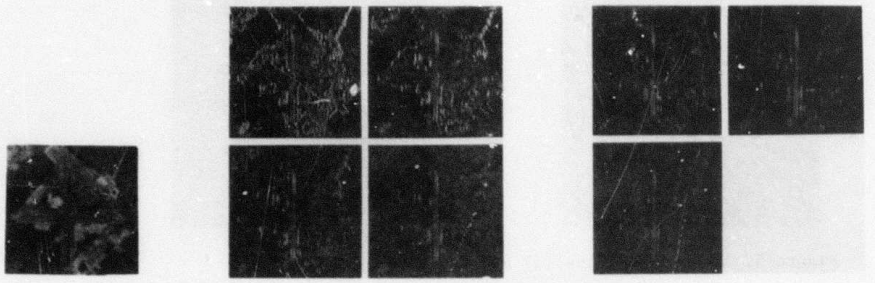
Figure 1. Brodatz's marble picture #62: (a) original; (b) thresholded at 25.



a. b. c1.

c2 c3

Figure 2. Three iterations of straight edge reinforcement applied to an aerial photograph of an airport. (a) Original; (b) Initial edge responses; (c) Results of iterations 1-3.



a. b1. b2.

Figure 3. Six iterations of strip enhancement applied to an aerial photograph of an airport. (a) Original; (b) Initial edge responses and results of iterations 1-6.

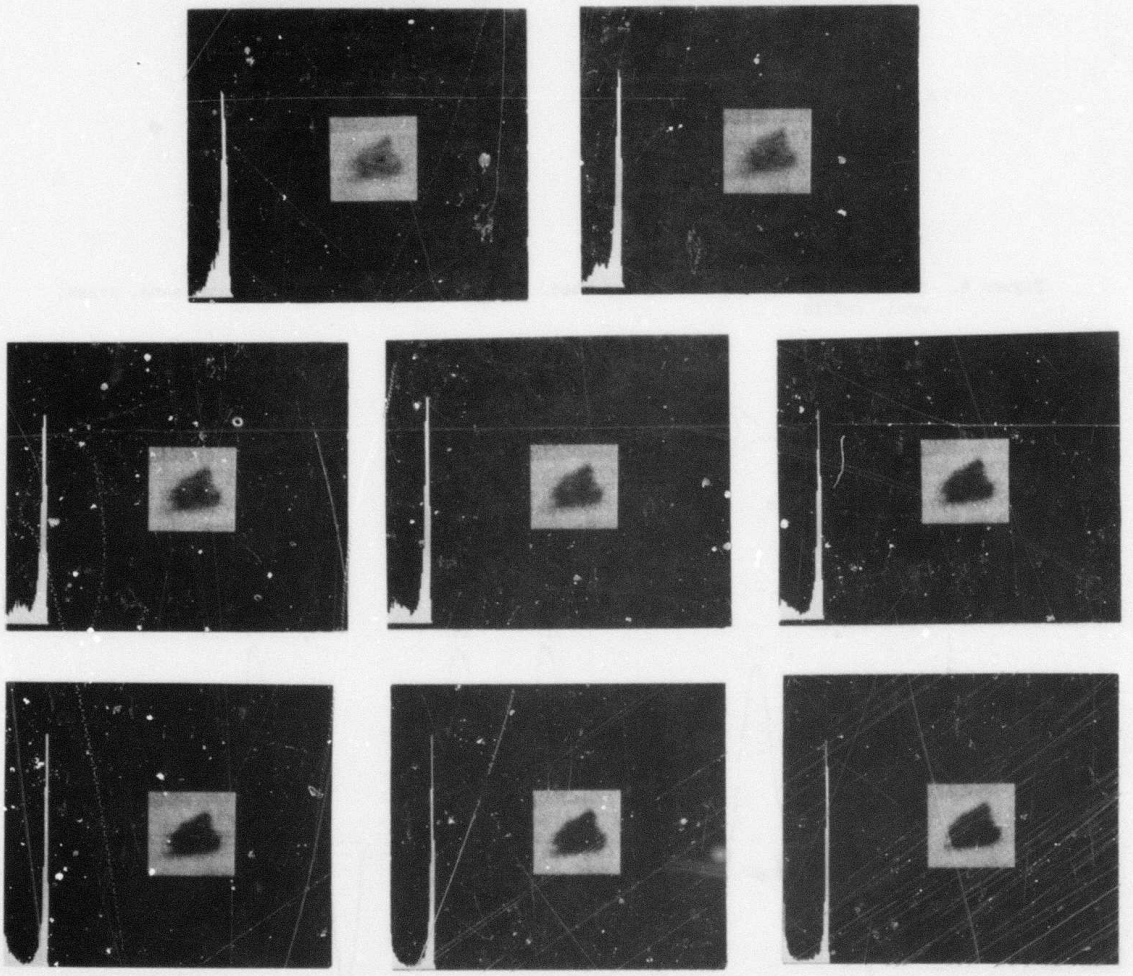


Figure 4. Thresholding by relaxation. The "light" probability of each pixel is displayed as a grey level. The parts show seven iterations and their corresponding histograms.

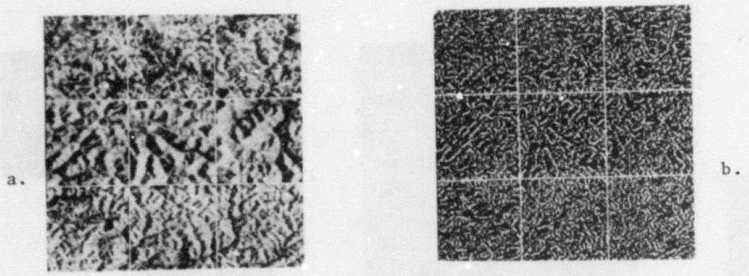


Figure 5. Terrain samples (a) and their edge maxima (b).

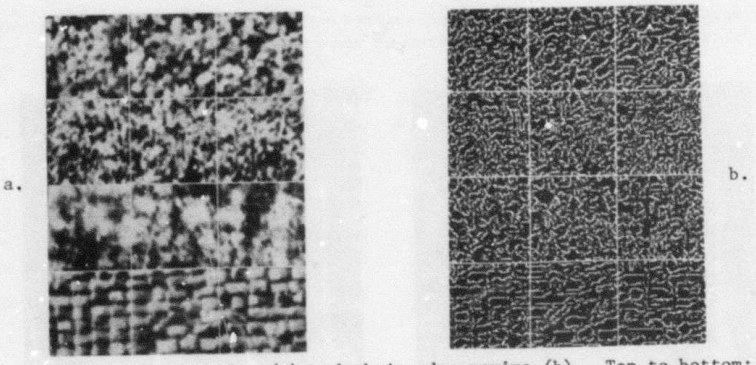


Figure 6. Brodatz texture samples (a) and their edge maxima (b). Top to bottom: sand, grass, wool, raffia.

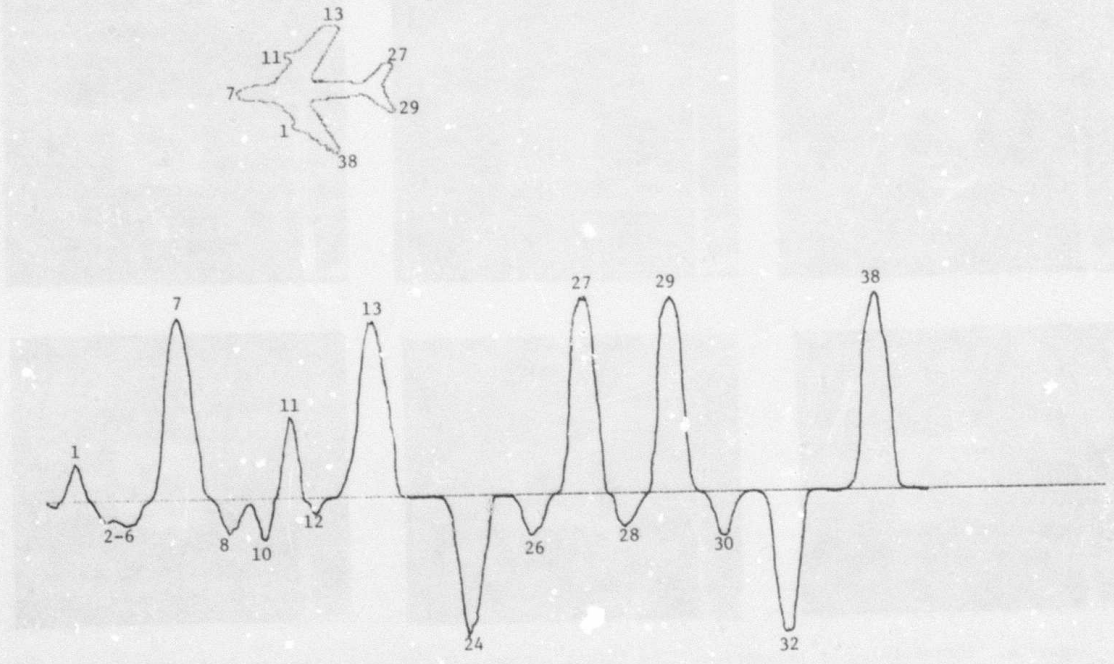


Figure 7. Airplane silhouette and its curvature plot, with peaks numbered.

MIT PROGRESS IN UNDERSTANDING IMAGES

Patrick H. Winston and the Staff

The Artificial Intelligence Laboratory
Massachusetts Institute of Technology

In this series of image understanding conference proceedings, we have stressed the key issue of representation. In particular, we have described the work of Horn and his collaborators using the reflectance map *and the* albedo image *in working with satellite images, and we have described the work of Marr and his collaborators using the* primal sketch, *the* 2 1/2-D sketch, *and* body-centered, 3-D models *to work toward a comprehensive theory of recognition.*

In the November, 1978 Proceedings, we reviewed the overall program, briefly explaining our approach, stating the objectives, and citing the fundamental tools. Then we summarized the results obtained through an enumeration of representative individual efforts.

Here we concentrate on Horn's group's work on hill shading *and* atmospheric modeling *and on Marr's group's discoveries about* texture *and about* zero-crossings, *with particular emphasis on* stereo.

Zero-Crossings and the Primal Sketch

The early work in Marr's group was largely devoted to the construction of a raw primal sketch of the image, a primitive description of the intensity changes in terms of edges, bars, blobs, and terminations, which are each characterized by position, orientation, contrast, and size. During the last year, the original structure of the primal sketch computation has been put aside, in favor of a better theory, because of recent developments in the theory of early vision. These are (i) the emergence of quantitative studies from psychophysics of the exact nature of the channels involved in early human vision [for example, Wilson and Bergen 1979], and (ii) Marr and Poggio's theory of stereopsis, which uses the zero-crossings in a set of second directional derivative operators for the stereo- matching process.

Computing the raw primal sketch falls naturally into two parts: (i) since intensity changes occur in natural images over a wide range of scales, we first detect and represent the intensity changes at a set of different scales; and (ii) the descriptions that arise from these independent channels are then combined into a single primal sketch of the image.

Marr and Hildreth have shown that provided some weak conditions are satisfied, intensity changes at a particular scale in an image I(x,y) are best detected by locating the zero-crossings of $\nabla^2 G(x,y) * I(x,y)$, where G(x,y) is the two dimensional Gaussian distribution, and ∇^2 is the Laplacian. The operator $\nabla^2 G$ uniquely satisfies certain critical properties of localization in space and frequency. The smallest operator usable in practice has a diameter of 9 picture elements in the central, positive region, with an overall support of roughly 1000 pixels. Interestingly, this is roughly the size of the smallest channel found in early human vision. The zero-crossings are then represented by a set of oriented primitives called zero-crossing segments, each describing a piece of the contour whose intensity slope (rate at which the convolution changes across the segment), and local orientation is roughly uniform. Small, closed contours are represented as blobs, also with an associated orientation, average intensity slope, and size defined by their extent along a major and minor axis.

Some intensity changes will give rise to zero-crossings over a range of adjacent scales, while others may be detected only at a single scale. In combining information from the separate channels, we take advantage of the observation that intensity changes in an image arise from surface discontinuities, or from reflectance or illumination boundaries, which all have the property that they are spatially localized. This observation led to the spatial coincidence assumption, which states that if similar information concerning the presence of an intensity change is found across a set of adjacent channels, they most likely describe the same physical intensity change, so their descriptions may be integrated into a single description of an edge. Information in one channel which does not coincide with that from adjacent channels is assumed to arise from a physical phenomenon which can only be measured at that one scale, so it gives rise to an independent descriptive element. The final raw primal sketch contains a binary map specifying the position of original zero-crossing contours, together with the symbolic description of the intensity changes, obtained from the separate channels. Figures 1, 2, and 3 illustrate these components of the raw primal sketch. Figure 1b shows the map of zero-crossing contours for the image in Figure 1a, while figures 2 and 3 show

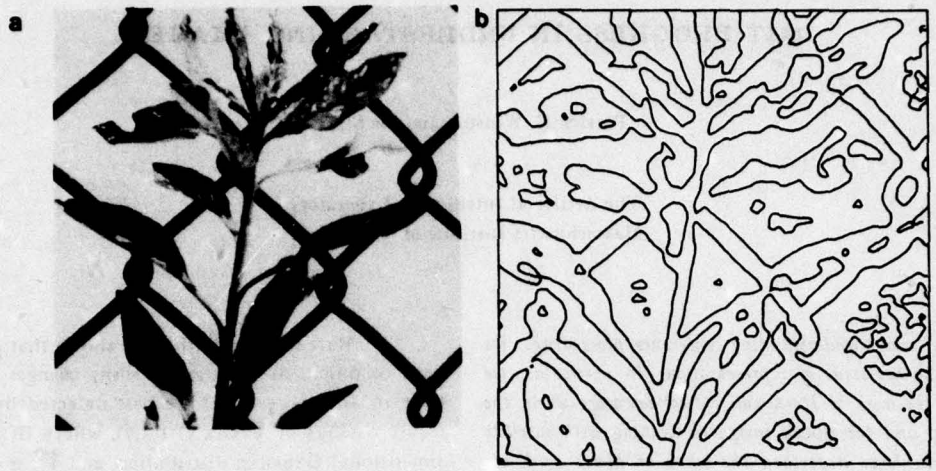


Figure 1: Zero-crossings in the output of the convolution of the above image with a $\nabla^2 G$ filter.

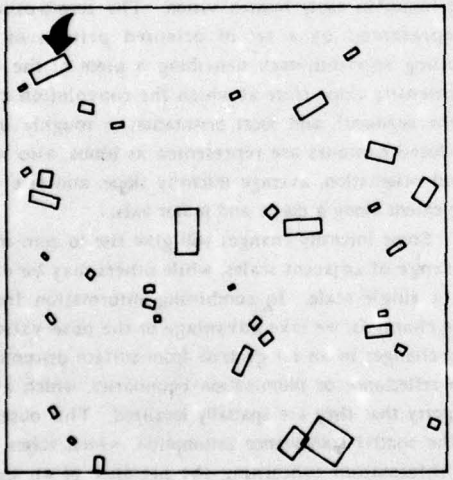


Figure 2: Symbolic representation of the blobs detected in the image. The arrow marks the blob whose full descriptor appears in the text.

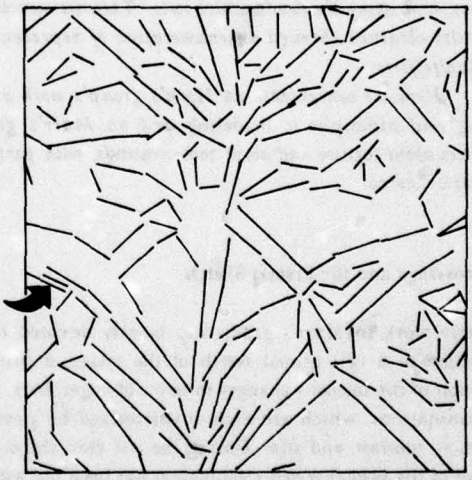


Figure 3: Symbolic representation of local orientations assigned to edge segments detected in the image. The arrow marks the blob whose full descriptor appears in the text.

symbolic representations of some of the descriptors attached to the locations marked in figure 1b. Figure 2 illustrates the blobs detected in the image, and figure 3 the local orientations assigned to edge segments. These diagrams show only the spatial information contained in the descriptors. Typical examples of the full descriptors are:

(BLOB (POSITION 146 21)
(ORIENTATION 105)
(CONTRAST 76)
(LENGTH 16)
(WIDTH 6))

(EDGE (POSITION 104 23)
(ORIENTATION 120)
(CONTRAST -25)
(LENGTH 25)
(WIDTH 4))

The descriptors to which these correspond are marked with arrows. The resolution of this analysis of the image roughly corresponds to what a human would see viewing it from a distance of about 6 feet. Finally, there is reason to believe that this representation is complete, and so, in principle, invertible [Marr Poggio and Ullman 1978].

Zero-Crossings and Primal-sketch Inversion

The new stress on zero-crossings raises the question of how complete is the information provided by them. For stereo analysis, the image is decomposed into a sum of approximately bandpass components (1-2 octaves wide at half-power points) and zero-crossing contours are obtained for each component. Marr, Poggio, and Ullman [1978] have suggested that each component may in fact be determined up to a constant factor by just its zero-crossing contours. By extending a theorem by Logan [1977] for the one-dimensional case, they showed this to be the case for two-dimensional entire functions with bandwidth less than an octave. However, it is not clear that this result would apply even approximately for practical signals which do not satisfy the ideal "less than an octave bandpass" requirement even at their half power points.

One way to relax the bandwidth restriction is to record the steepness of the surface (surface gradient) through the zero-crossings as well. To investigate the information content of zero-crossing contours and the surface gradient along those contours in practical situations, the problem of reconstructing the original surface from that information was studied.

A direct approach to reconstruction would be to find an interpolation operator which could be used to reproduce the boundary conditions given. These are the approximate bandpass characteristic and the zero-crossing contours and intensity slopes. Thus one wants an interpolation function that preserves the bandpass property while allowing the introduction of zero crossings with a given gradient. Perhaps the simplest suitable local operator is the derivative of the two-dimensional Gaussian:

$$\frac{dg}{dx} = -2x\,e^{-(x^2+y^2)}$$

This function dies away rapidly with distance from the origin and its only zero-crossing contour lies on the y-axis and has maximum gradient at the origin. The Fourier transform of this function has the same form and so dg/dx has a half power bandwidth slightly greater than that obtained with the difference of Gaussian (DOG) convolution. The same holds for translations and rotations of dg/dx in the x, y plane, and for sums of these functions. Thus it is possible to construct a function which has a given set of zero-crossing contours and gradients along those contours with arbitrary accuracy with a sum of the form:

$$F(x,y) = \sum_{i=1}^{N} [a_i(x-x_i) + b_i(y-y_i)]\; e^{-\frac{(x-x_i)^2+(y-y_i)^2}{\sigma^2}}$$

For example a rough approximation to the surface zero-crossing and zero-crossing gradient is obtained if N evenly spaced points $(x_i\ y_i)$ are selected along the zero-crossing contours and $(a_i\ b_i)$ is the gradient (df/dx df/dy) of the original surface at the zero-crossing point $(x_i\ y_i)$. The fixed constant σ determines the bandpass range of the sum. Since the individual terms of the sum above are small everywhere except in the immediate neighborhood of $(x_i\ y_i)$ and the terms themselves are spaced evenly, the gradient of the sum in the vicinity of each point $(x_i\ y_i)$ is determined largely by the ith term. This makes it possible to design a simple iterative algorithm for adjusting the ith term so that the position and gradient of the zero-crossing achieve the desired value at $(x_i\ y_i)$. It turns out that a two-dimensional function which satisfies the zero-crossing boundary conditions, and which has a similar spatial frequency bandpass, can be constructed rather efficiently by this means. Such functions can be used to test empirically conditions under which the zero-crossing information is sufficient to determine the overall function.

Figures 4 through 7 show the results of the above reconstruction method applied to the zero-crossing gradients of an image convolved with a DOG mask with central panel width of 12 pixels. Figure 4 shows the positive values of the original 128 by 128 convolved image as gray levels.

The negative values and values near zero appear as white. The x's indicate the zero-crossing points used for the reconstruction which were obtained by dividing the image into a grid of 6 by 6 squares and selecting one zero-crossing point from each square having a zero-crossing contour through it. Figure 5 shows the surface reconstructed using just the surface gradient at each of the x's. Figures 6 and 7 are overlays of graphs of horizontal slices through the original surface in figure 4 and the reconstruction in figure 5.

Further work is planned to explore the possibility of using similar techniques to approximate a function from just its zero-crossing contours.

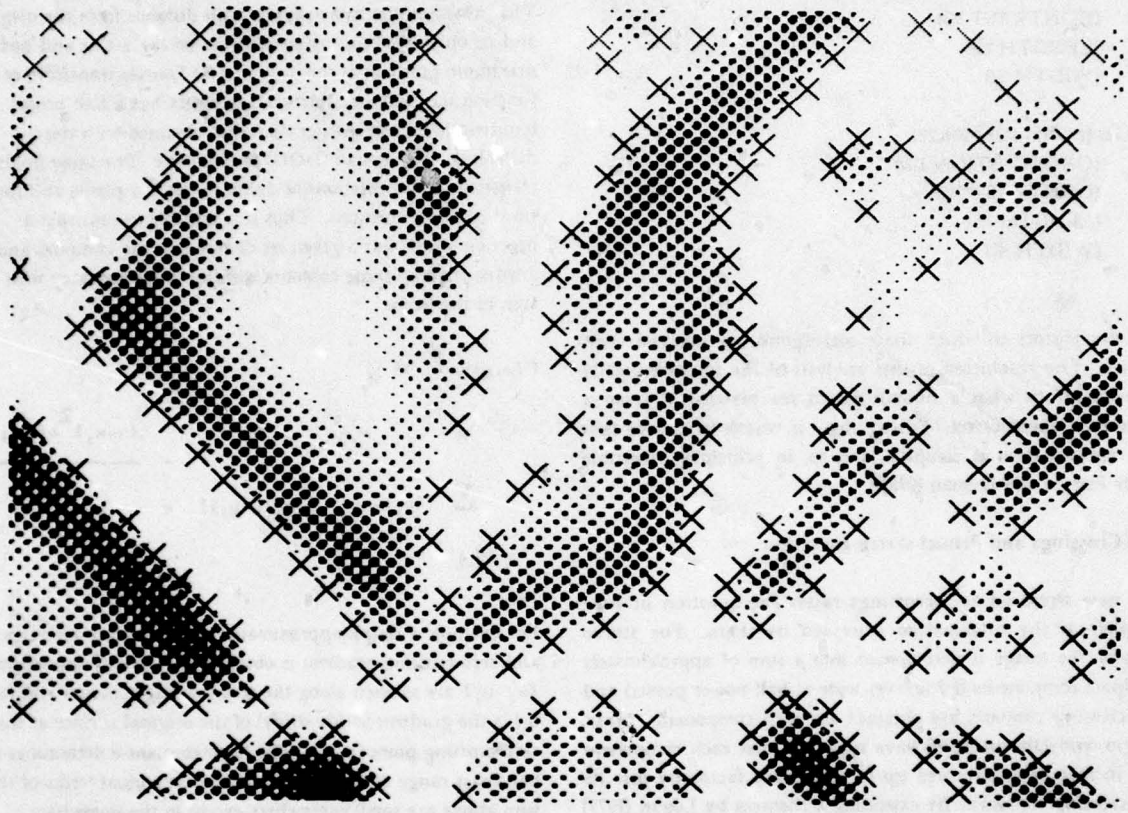


Figure 4: Image convolved with a difference of Gaussian mask.

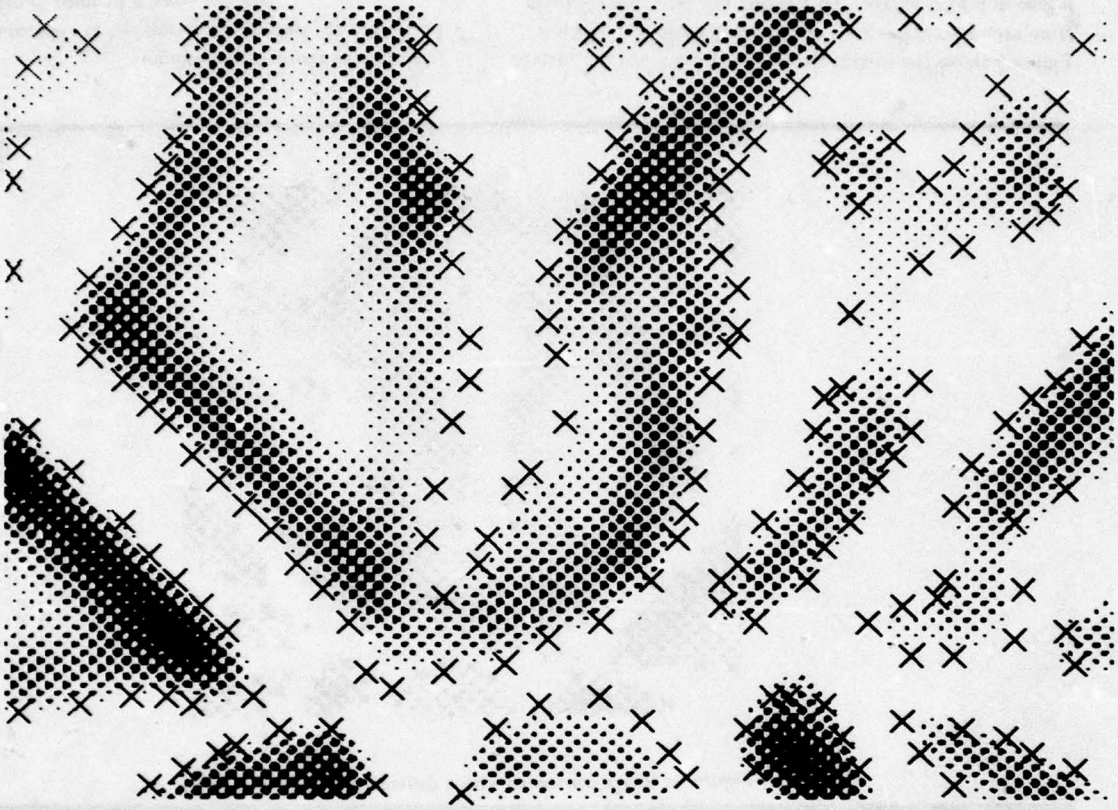


Figure 5: Image reconstructed from zero-crossings information.

The negative values and values near zero appear as white. The x's indicate the zero-crossing points used for the reconstruction which were obtained by dividing the image into a grid of 6 by 6 squares and selecting one zero-crossing point from each square having a zero-crossing contour through it. Figure 5 shows the surface reconstructed using just the surface gradient at each of the x's. Figures 6 and 7 are overlays of graphs of horizontal slices through the original surface in figure 4 and the reconstruction in figure 5.

Further work is planned to explore the possibility of using similar techniques to approximate a function from just its zero-crossing contours.

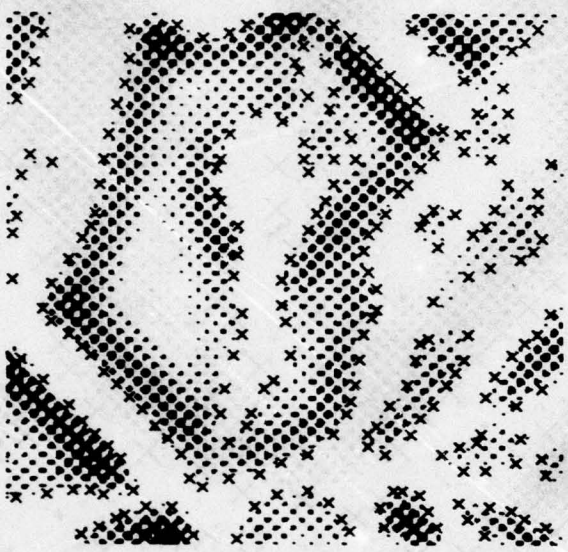


Figure 4: Image convolved with a difference of Gaussian mask.

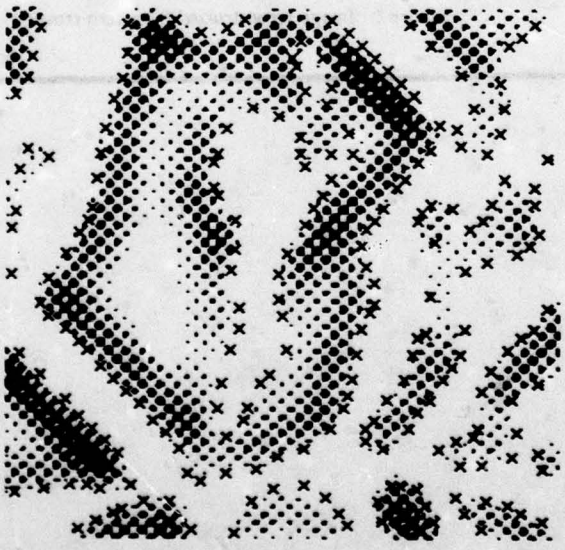


Figure 5: Image reconstructed from zero-crossings information.

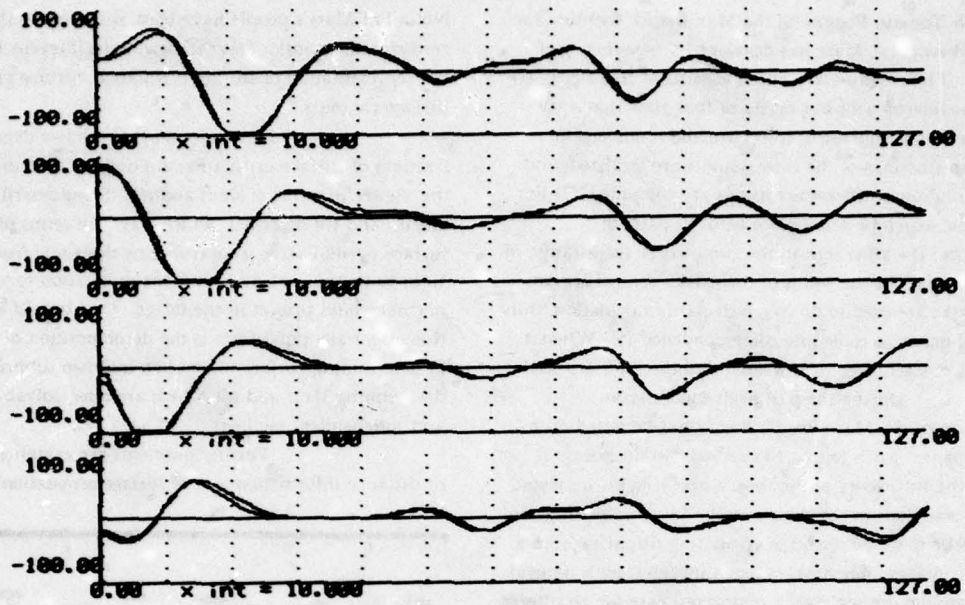


Figure 6: Overlays of slices through the surfaces shown in figures 4 and 5. Slices are parallel to the x axis at y = 32, 40, 48, and 56.

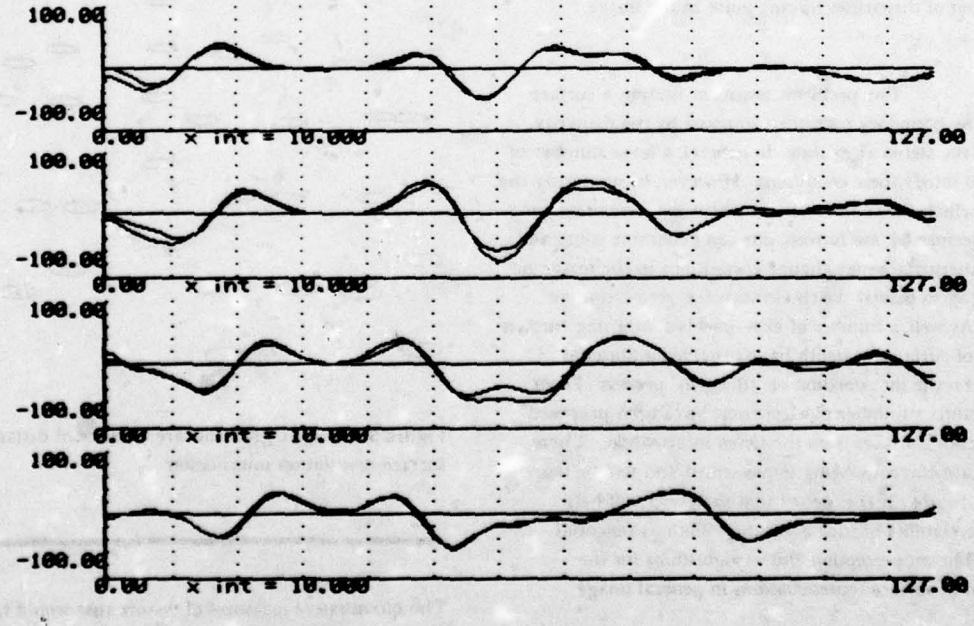


Figure 7: Additional overlays at y = 64, 72, 80, and 88.

Zero-Crossings and Stereo Theory

Working with Tomaso Poggio (of the Max Planck Institute for Biological Cybernetics), Marr has developed a new theory of stereo vision. The resulting algorithm consists of five steps: (1) Each image is filtered with bar masks of four sizes that vary with eccentricity; the equivalent filters are about one octave wide. (2) Zero-crossings of the filter outputs are localized, and the local orientation of the zero-crossings are computed; (3) For each mask size, matching takes place between pairs of zero-crossings of the same sign in the two images, for a range of disparities up to about the width of the mask's central region; (4) Wide masks are used to do rough disparity calculation, thus causing small masks to come into correspondence; (5) When a correspondence is achieved, it is written into the 2 1/2-D sketch.

During the past twelve months, an implementation of the Marr-Poggio theory has be tested on a number of images. Such testing has served two purposes: it helps prove the sufficiency of the theory, and it helps indicate problems or omissions in the theory itself. The implementation was found to be quite successful in computing disparity from a stereo pair of images. The images tested included both natural scenes and random dot patterns, a critical test case for any stereo program.

The nature of the theory underlying the stereo algorithm, and in fact the nature of the current theory of the primal sketch, require that th stereo algorithm will at best only determine disparity values along certain contours in the image, the primal sketch descriptors. A natural next step is the determination of disparities for any point in the image.

The problem consists of finding a surface which fits the boundary conditions imposed by the disparity contours of the stereo algorithm. In general, a large number of surfaces will satisfy these conditions. However, by analyzing the process by which the contours, upon which the stereo-matching process is performed, are formed, one can determine constraints upon how the surface may change at positions in the image not corresponding to primal sketch elements (i.e. zero-crossings contours). As well, a number of experiments concerning human perception of surfaces in depth have provided additional constraints on the interpolation or "filling in" process. From such constraints a number of algorithms have been proposed for constructing surfaces from the stereo information. These algorithms are currently being implemented and will be tested in the near future. It is expected that such tests will help indicate the viability of such algorithms, both as potential models for human perception and as algorithms for the construction of surface representations in general image processing.

Texture

Not all of Marr's people have been absorbed by the zero-crossings work. Stevens' new thesis [Stevens 1979] addresses the representation of surface orientation, texture gradients, and surface contours.

He has found that the two degrees of freedom of surface orientation are usefully described relative to the viewer in terms of *slant* and *tilt*. Slant describes "how much," and tilt describes "which way." In terms of the local surface normal, slant is measured by the angle from the line of sight to the normal, and tilt is the orientation to which the normal would project in the image. One benefit afforded by this essentially polar form is the decomposition of the problem of determining surface orientation into two subproblems, determining slant and tilt, which are often solvable by distinct and independent methods.

Texture gradients are examined as a source of distance information and of surface orientation. See figure 8

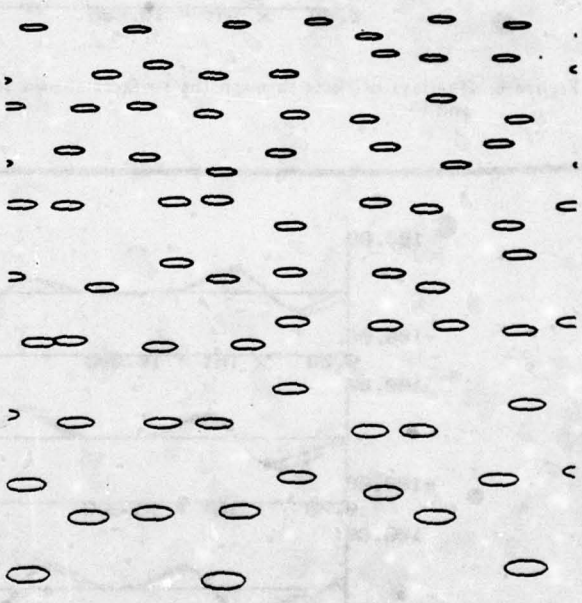


Figure 8: Texture gradients are a source of distance and surface-orientation information.

The quantitative measures of texture that would be necessary for either computation are naturally described relative to the *local tilt frame*, a local coordinate system whose y-axis is aligned with the texture gradient, and therefore corresponds to the

surface tilt. The tangent of the slant angle is proportional to the rate of change of texture density, area, and lengths measured relative to the tilt frame. But distance (up to a scale factor) can be simply computed by the reciprocal of lengths measured along the x-axis, assuming that those lengths correspond to uniform surface texture dimensions. (Texture lying along the x-axis corresponds to surface texture that is equidistant from the viewer -- those dimensions are not foreshortened.) Stevens has observed that the orientation of the x-axis corresponds to the image orientation in which texture is locally constant, that is, the tilt frame can be determined without considering the orientation of the gradient. Hence a "depth map" may be derived from a "texture gradient" without either determining the orientation of the gradient or its magnitude. Surface shape is therefore more simply computed in terms of distance than local surface orientation.

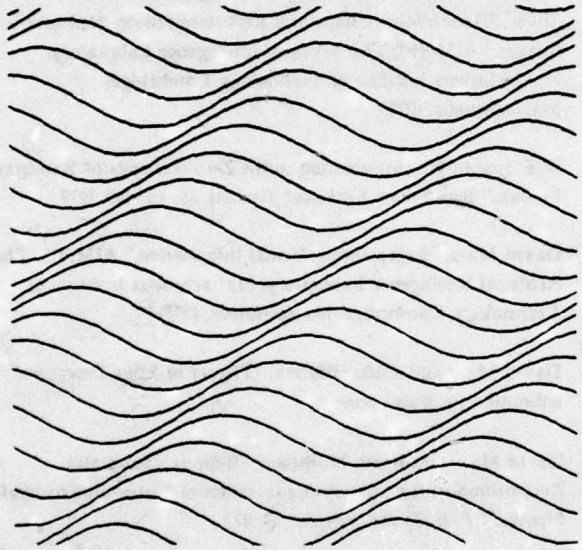


Figure 9: Surface contours yield information about shape.

Surface contours are an important source of information about surface shape, however their analysis has received almost no theoretical attention. See figure 9. The edge of a shadow cast on a surface, for example, may tell us of the shape of the surface, given some assumptions. The contours in the image of glossy surfaces are similarly useful. But in order that a surface contour may tell us of the shape of the surface, various assumptions must be made. To explore these assumptions, Stevens has decomposed the problem into two aspects: (a) determining the shape of the three-dimensional curve that Marr calls the *contour generator*, whose image is the surface contour, and (b) determining how the surface lies under the contour generator. This is analogous to (a) bending a wire in space, so that it appears like the surface contour, and (b) gluing a ribbon along the wire, the ribbon representing the strip of surface under the contour generator. The first aspect may be constrained by assuming the principle of general position, planarity, symmetry, and constancy of curvature along the contour generator. The second aspect may be constrained by assuming either that the contour generator is a planar, asymptotic curve (as are gloss contours in orthographic projection) or a geodesic curve. In either case, the surface orientation can be solved once the three-dimensional shape of the contour generator is determined.

Making Faithful Synthetic Images Requires Good Atmospheric Modeling

Turning now to Horn's work, recall that in the last proceedings, we listed the following uses for synthetic images: automated generation of shaded relief maps, generation of low-level, obliquely-viewed images, generation of special maps that bring out particular terrain features, classification of ground cover for crop prediction, matching images to terrain data for satellite navigation, and making maps for automatic or semiautomatic change detection.

In general, four factors must be considered when making synthetic images for these purposes. They are: (i) imaging geometry - the projection of the viewed scene onto the image, (ii) incident illumination - the intensities and distribution of light sources, (iii) surface photometry - the way a surface reflects light, and (iv) surface topography - the shape of things in the scene.

Synthetic images that are to mimic obtained from spacecraft require attention to a fifth factor: the atmosphere attenuates visual signals, scatters spurious light into the viewing port of the satellite, and illuminates the ground as a large, diffuse light source.

If effects of imaging geometry, illumination, topography, and atmosphere are removed, then ground cover can be identified. Previous MIT research has lead to understanding the geometry of LANDSAT imaging [Horn and Woodham 1978], and to success in eliminating topographic effects and differential exposure to direct sunlight [Horn 1978]. The next step is to consider carefully the nature and extent of the sky's influence.

The problem of atmospheric effects is quite complex and mathematically intricate. The literature of remote sensing, atmospheric science, geophysics, and space science is filled with detailed reports on absorption, transmission, and radiance, complete tabulations of wavelength-dependent behavior, and sophisticated models of particle scattering. However, the emphasis has always been on the degradation of visual signals passing through the atmosphere. The little

research on the contribution of sky luminance to the imaging process has been restricted to flat areas of the earth's surface, either agricultural lands or the ocean. Woodham and Horn [1978] have shown that, at least in the shorter wavelengths, sky illumination is not negligible in LANDSAT multispectral scanner images. Part of this is due to the relatively low sun elevation (about 9:30 a.m. local time for LANDSAT 1 and 2) which puts a large portion of rugged terrain in shadow. For these areas, the sky is the only light source (ignoring such things as mutual illumination of one side of a valley by the opposite side). Work is underway to investigate the interaction of sky radiance and surface reflectance in areas of rugged topography.

The research approach follows that of prior work in the production of albedo maps. A high-resolution digital terrain model represents the topography of a section of the earth. A synthetic image [Horn and Bachman 1978] is created, assuming uniform Lambertian surface reflectance, a point sun at a given elevation and azimuth, known LANDSAT imaging geometry, and a model of sky luminance, absorption, and backscatter. Point-by-point comparison with actual LANDSAT imagery of the corresponding earth region shows variations from the assumed uniform reflectance, from which the intrinsic reflectance, or albedo, of each point is calculated. A new picture (the albedo map) is created from these albedos, which is suitable for terrain classification. The problem, of course, is to develop a sufficiently accurate model of the atmosphere. The model must be computationally feasible, the simpler the better. Therefore, it must use as much local information as possible, that is, be applicable to individual pixels and their neighborhoods. By their very nature, however, atmospheric effects are global. Some theoretical work has already been done relating general illumination and surface reflectance [Horn and Sjoberg 1979]. Work is continuing.

REFERENCES

For an extended discussion of MIT work on Image Understanding, see Volume 2 of *Artificial Intelligence: an MIT Perspective*, edited by Patrick H. Winston and Richard H. Brown, MIT Press, Cambridge, Massachusetts, 1979.

W. E. L. Grimson and David Marr, "A Computer Implementation of a Theory for Human Stereo Vision, included in these proceedings.

Berthold K. P. Horn, "The Position of the Sun," Working Paper 162, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978.

Berthold K. P. Horn and Brett L. Bachman, "Using Synthetic Images to Register Real Images with Surface Models," AIM-437, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977. Also in *CACM*, November, 1978.

Berthold K. P. Horn and Robert W. Sjoberg, "Calculating the Reflectance Map," AIM-495, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978. Also to appear in *Applied Optics*, June, 1979.

Berthold K. P. Horn and Robert J. Woodham, "LANDSAT MSS Coordinate Transformations," AIM-465, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978.

Berthold K. P. Horn and Robert J. Woodham, "Destriping Satellite Images," AIM-467, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts 1978. Also to appear in *CGIP*.

Berthold K. P. Horn, Robert J. Woodham, and William M. Silver, "Determining Shape and Reflectance using Multiple Images," AIM-490, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978.

B. F. Logan Jr., "Information in the Zero-crossings of Bandpass Signals," *Bell System Technical Journal*, 56, 487-510, 1977.

David Marr, "Representing Visual Information," AIM-415, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

David Marr and Ellen Hildreth, "Theory of Edge Detection," *submitted for publication*.

David Marr and Keith Nishihara, "Representation and Recognition of the Spatial Organization of Three-dimensional Shapes," *Phil. Trans. Roy. Soc. B. 275*.

David Marr and T. Poggio, "A Computational Theory of Human Stereo Vision," AIM-451, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978. Also to appear in *Proc. R. Soc. Lond. B.*

David Marr, T. Poggio, and S. Ullman, "Bandpass Channels, Zero-crossings, and Early Visual Information Processing," AIM-491, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978.

Kent Stevens, "Surface Perception from Local Analysis of Texture and Contour", Ph. D. Thesis, February, 1979. To appear as TR-512, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Thomas M. Strat, "Shaded Perspective Images of Terrain," AIM-463, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978.

Thomas M. Strat, "Application of Data Flow Computation to the Shaded Image Problem," Working Paper 163, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978.

H. R. Wilson and J. R. Bergen, "A Four Mechanism Model for Spatial Vision," (in the press).

Robert Woodham and Berthold K. P. Horn, "Looking in the Shadows," Working Paper 169, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1978.

PROGRESS AT THE ROCHESTER IMAGE UNDERSTANDING PROJECT

C. M. Brown
J. A. Feldman
K. R. Sloan, Jr.

Computer Science Department
The University of Rochester
Rochester, New York 14627

1. Model Refinement

1.1. Procedural Description

One important goal of the Rochester Vision Project is to investigate a generalized form of procedural invocation in which an executive procedure chooses worker procedures to perform a job not just on the basis of input/output behavior (as traditional pattern- directed invocation does), but also taking into account cost/benefit estimates and perhaps other information as well. This scheme is motivated by the desire to have the advantages of declarative knowledge about what is doable (the descriptions) along with the advantages of procedural knowledge about how to do it (the workers). The declarative, descriptive component will allow conviences such as the modular addition of procedural knowledge. The main research issue is to decide what exactly needs to be known about worker procedures, and how to express that in a useful and uniform manner. This must also be coordinated with the use of relational constraints [Russell and Brown, 1978]. The most recent and presently contemplated work at Rochester explores aspects of these issues (e.g. Lantz, Ballard, and Brown, 1978).

1.2. Decision Theory

The use of decision theory not only as an abstract model of intelligent perception but as a practical tool to maximize computational benefit/cost is being investigated in the context of procedural invocation. This work continues in the tradition of Bolles, Sproull, and Garvey, and ultimately we hope to extend some of their results to deal with formal problems that more closely approximate the sorts of vision problems encountered in our particular applications. Ballard (see Section 2) uses decision theory techniques to choose the most economical method (assuring adequate accuracy) of locating anatomical structures in large-format images.

2. Applications in Biomedicine

The model-directed finding of ribs in chest radiographs [Ballard, 1978] provides an illustration of the use of the Rochester Vision System, incorporating procedure description, utility measures, and tops-down, model-directed perception. The object here is to cope with large amounts o. possibly low-quality data without undue processing time by depending on a declarative model of anatomical structures, described procedural knowledge about how to locate them, and an executive which uses decision theory to control the image- understanding process. A prototype complete analysis system is now being developed.

A novel and uniform method of describing arbitrary functions on the unit sphere (which define "museum-viewable" volumes) is under investigation, with immediate application to anatomical structures [Schudy 1978]. The idea is related to the well- known Fourier descriptions of two- dimensional shape. Volumes are modelled and described as the leading coefficients in certain spherical harmonic expansions of the volume functions. This method also allows least squared error fitting of volumes in coefficient space, which interfaces nicely with routines which locate the three- dimensional boundaries of volumes in image data.

3. Application in Aerial Image Analysis

The three-level organization of image analysis (strategist, executive, worker) and a further exploration of useful procedural description mechanisms are the objects of study in automatic photo- interpretation work [Lantz 1978]. The object is to use the sorts of knowledge- based inferencing used by skilled photointerpreters, along with models inspired by photointerpretation keys for identifying small industries, to do reliable and flexible

identification of a few types of small industrial installations. Imagery has been acquired from a Rochester, N.Y. mapping firm and from RADC in Rome, N.Y.

4. Image Encoding and Transmission

4.1 Hierarchical Image Encodings

Communication of images, and information about images is an important part of any image understanding project. We have been investigating the use of various hierarchical image encodings. One of the image transmission schemes we have investigated is closely related to "pyramid" data structures. We have demonstrated that high resolution raster images can be effectively transmitted over relatively low-bandwidth lines by sending a series of low resolution approximations, which converge to the final image [Sloan and Tanimoto, 1978].

A second hierarchical encoding is described elsewhere [Ballard, 1979] in these proceedings. A Strip Tree is a simple encoding for linear features. Efficient algorithms have been developed to take advantage of the hierarchy.

4.2 Composition and Re-interpretation of Images

It is often convenient to specify an image in terms of the combination of several existing images, rather than transmit an entire new image. The combination or re-interpretation may sometimes be performed with relatively simple hardware devices. We have developed and implemented several such techniques based on the "video lookup table" supplied with our Grinnell GMR-26 display [Sloan and Brown, 1979]. These techniques are currently being used to overlay map features on aerial images, display three-dimensional surfaces under quickly varying lighting conditions, and show short, repetitive motion sequences.

5. Component Building

5.1. Hardware

The Grinnell GMR-26 display device is on site and DMA-interfaced to an Eclipse computer. 32K of core has been added to the Vision Eclipse, which is also used for research in distributed computing (see Section 5.2). The original 80MB disk has been replaced with a 300MB one, and another 300MB disk has also been installed along with a much faster controller, leading to greatly enhanced performance. We are acquiring terminals and investigating how to meet our everyday computing needs by commercial, home-built, or combination intelligent terminal systems. An Optronics Colorscan C-4100 drum scanner is on site and interfaced to the Vision Eclipse. The fast (50KB) link to the PDP-KL10 has been completed and is operating well.

5.2. Software

Advanced system software support is now used routinely, and more is under development. Communications protocols and distributed computing packages [Rovner 1978, Feldman 1978, Sheininger and Sabbah 1978, Selfridge 1978, Sloan 1978] have been developed to allow access to the GMR-26 through the local ALTO computers or the remote PDP-10, to achieve reliable transmission between distributed processes, to produce graphics and halftone images on ALTO screens from the PDP-10, and to allow file transfer and telnet to the Arpanet. The IPCF in the TOPS-10 operating system is the basis for communication between PDP-10 jobs, and these jobs may now create RIG messages and send them to the local operating system for disposition. At Rochester, the RIG message is the lingua franca that allows processes on remote machines to command the GMR-26, perform file manipulations, and other operations. Some of our work has been utilized by other image understanding groups, most extensively at SRI. Some student projects in our Computer Vision courses are aimed at producing useful system software for vision, and the common departmental interest in distributed computing assures that new and co-operative efforts using the distributed computation and communications packages will be launched frequently. A comprehensive library of vision routines [Sloan 1977-78] has been developed, centralized, documented, and incorporated into the NEXUS system. They allow interactive users a wide range of image-processing and display (graphics, halftone, color and B&W TV) capabilities. A program to acquire images from the Optronics scanner and package them according to our Raster Image File Format [Selfridge and Sloan, 1979] has been developed and is in routine use.

6. Motion Understanding

Understanding motion pictures has always presented an unusually difficult problem to computer vision efforts. The compelling gestalt induced in humans by moving objects is not well understood, and so there is little leverage on the immediate problems resulting from the large mass of data in multi- frame images. We are hoping to make progress first on a pared-down version of the problem which nevertheless offers an

interesting set of perceptual phenomena to model. The domain is multi- frame images of animal motion; initial research is being carried out on sequential images of points of light attached to joints. This data can give humans a strong perception of coherent motion, and present work is aimed at understanding how we correctly identify points (about 13 in all in present data) from frame to frame, and how we segment the resulting moving points into meaningful body parts. Ultimately, the results will be applied to multi-frame grey-scale images. Data presently comes from a program which simulates a range of human walking motion in 3-D. The program is a useful theoretical tool, since it allows direct access (not mediated by vision) to movement parameters, and point locations. It is also a useful psychological research tool, since with it one can inexpensively investigate limits in human performance.

7. Texture

Textural areas can be thought of as those parts of an image where segmentation based on normal similarity measures fails. Meaningful analysis of textured areas must include discrimination between different textures and detection of parts of the same texture. The similarity of textures which are identical except for a scale change, a rotation, or a different range of intensities must be recognized.

We approach the texture problem by dividing texture regions into meaningful sub-elements of similar intensity sample points, then using rotation- and scale-invariant shape measures to characterize these regions and finally determining spatial relationships among our sub-elements. By using a decision tree program structure, easily discriminated textures are separated quickly, and more complex textural structure is extracted only when necessary [Maleson, 1978].

8. Programming Language Development

The Smart Compiler and Distributed Computation research groups are cooperating on a language for research into both these fields [Ball 1978]. It will contain the ideas of PLITS, together with improvements and extensions gleaned from the SAIL-PLITS implementations of the past. There are several separate ways in which the programming language developments are affecting Image Understanding research in our laboratory and elsewhere [Feldman & Williams 1977; Feldman, 1978]. Many of the ideas developed in this work are being heavily used in image understanding.

REFERENCES

Ball, J.E., et al., ZENO: a language for smart compiler research, Internal Memo, Computer Science Department, University of Rochester, 1978.

Ballard, D.H., Model-directed detection of ribs in chest radiographs, TR11, Computer Science Department, University of Rochester, March 1978.

Barrow, H.G., et al., Interactive aids for cartography and photo interpretation, Semiannual Technical Report, Artificial Intelligence Center, SRI International, November 1977.

Brown, C.M., Fast display of certain museum-viewable polyhedra, TR23, Computer Science Department, University of Rochester, March 1978.

Feldman, J.A., Synchronizing distant cooperating processes, TR26, Computer Science Department, University of Rochester, October, 1977.

Feldman, J.A., Systems support for advanced image understanding. DARPA Semiannual Technical Report, May 1978.

Feldman, J.A., and Williams, G. Some comments on datatypes, TR28, Computer Science Department, University of Rochester, December 1977.

Lantz, K.A., Procedural knowledge and control in a model - driven vision system, Thesis proposal, University of Rochester, February 1978.

Lantz, K.A., Ballard, D.H., and Brown, C.M. General invocation through procedure descriptions: two applications in image analysis, 22nd International Symposium of the Society of Photo-optical Instrumentation Engineers, San Diego, CA., August 1978.

Rashid, R.F. Motion understanding, Thesis proposal, University of Rochester, in preparation 1978.

Rovner, P.D. Flow control and reliable transmission in a system for distributed computing, TR22, Computer Science Department, University of Rochester, October 1977.

Rovner, P.D. Automatic representation selection for associative data structures, to appear in Proceedings of the National Computer Conference, Anaheim, CA., June 1978.

Russell, D.R., and C.M. Brown, Representing and using locational constraints in aerial imagery, Image Understanding Workshop, November, 1978.

Sabbah, D. Image calibration, Internal Memo, Computer Science Department, University of Rochester, in preparation 1978.

Scheininger, U., and Sabbah, D., The display process, Internal Memo, Computer Science Department , University of Rochester, December 1977.

Schudy, R., A model for echocardiography, TR12, Computer Science Department, University of Rochester, (in preparation) 1978.

Selfridge, P., Name - value pairs in the Rochester vision header. Internal Memo, Computer Science Department, University of Rochester, January 1978.

Selfridge, P. and Sloan, Jr., K.R., "Raster Image File Format (RIFF): An Aproach to Problems in Image Management," TR 52, Computer Science Department, University of Rochester, May, 1979.

Sloan, Jr., K.R., Rochester vision library documentation, Internal Memos, Computer Science Department, University of Rochester, 1977 - 1978.

Sloan, Jr., K.R., and Brown, C.M., "Color Map Techniques," to appear in CG and IP.

Sloan, Jr., K.R., and Tanimoto, S.L., "Progressive Refinement of Raster Images," TR39, Computer Science Department, University of Rochester, November. 1978.

Wellner, J.A., Two-sample tests for a class of distributions on the sphere, submitted for publication, February 1978.

SESSION II

TECHNICAL PAPERS

A computer implementation of a theory of human stereo vision

W. E. L. Grimson and D. Marr

M.I.T. Artificial Intelligence Laboratory,
545 Technology Square, Cambridge MA 02139, U.S.A.

Introduction

In a recent article, Marr & Poggio (1979) set out a computational theory of human stereo vision. According to this theory, the human visual processor solves the stereoscopic matching problem by means of an algorithm that consists of five main steps: (1) The left and right images are each filtered at different orientations with bar masks of four sizes that increase with eccentricity; these masks have cross-section that is approximately the difference of two gaussian functions, with space constants in the ratio 1:1.75. (2) Zero-crossings in the filtered images are found, along scan-lines lying perpendicular to the orientation of the mask. Termination points of lines and edges are also localized. (3) For each mask size, matching takes place between pieces of zero-crossing of the same sign and roughly the same orientation in the two images, for a range of disparities up to about the width of the mask's central region. Within this disparity range, Marr & Poggio showed that false targets pose only a simple problem. (4) The output of the wide masks can control vergence movements, thus causing small masks to come into correspondence. In this way, the matching process gradually moves from dealing with large disparities at low resolution to dealing with small disparities at high resolution. (5) When a correspondence is achieved, it is stored in a dynamic buffer, called the $2\frac{1}{2}$-dimensional sketch.

There are several reasons why it is important to implement a computational theory like this one. Firstly, it serves as a means of testing the sufficiency of the theory. That is, by running the program on various pairs of stereo images, one can examine the performance of the program, and hence of the theory itself, provided the program is an accurate representation of that theory. Secondly, such an implementation serves as a useful feedback device, enabling one to test the critical factors of the theory and to illuminate errors or omissions of the theory. And thirdly, it enables one to assess the validity of assumptions made by the theory, concerning for example the statistical structure of images.

This article describes an implementation of the stereo theory that was written with particular emphasis on the matching process. We first set out the overall design of the program, and then we show two examples and assess the program's overall performance on various types of image.

Design of the program

The implementation is divided into five modules, roughly corresponding to the five sections in the summary that we gave above. We describe each in turn.

I Input and convolution

The input to the program consists of a stereo pair of images, digitized on an Optronix Photo-reader. The sizes of these images are 320 X 320, and grey-level resolution is 8 bits (256 grey levels).

The main difference in this first stage between our program and the original description of the theory is that the initial convolutions are performed with non-oriented filters. This is possible, because Marr & Hildreth (1979) showed that under rather weak conditions, the zero-crossings obtained with the Laplacian were equivalent to those obtained with a set of second directional derivatives. Although in theory, the optimal filter is $\nabla^2 G$, where ∇^2 is the Laplacian and G is a two-dimensional Gaussian distribution we have used its approximation by radially symmetric differences of Gaussians, with space constants in the ratio 1:1.75, (after Wilson & Bergen 1979).

There remains only one free parameter to specify the filter completely--its overall size, which is conveniently specified by the width w of the filter's central excitatory region. Wilson & Giese's (1978) data indicated values of w for the central fovea of the human visual system in the range 3' to 12' of visual arc. If one considers the experimental conditions under which Wilson obtained his data, it becomes apparent that these measurements essentially correspond to a projection of the two-dimensional mask onto a line. If one projects a radially symmetric difference of gaussians (or DOG) filter onto a line, one obtains a one-dimensional DOG with slightly smaller w--in fact $w_{1\text{-}DM} = w_{2\text{-}DM}/2$.

Taking this into account, and using the figure of 20"-30" for the separation of cones in the fovea (see Cornsweet 1970 p. 356), one arrives at values of w in the range 9 to 35 pixels. In our program , three initial filters are used, with w values of 9, 17 and 35 pixels. The supports of these masks are roughly 1000, 3600 and 15200 pixels respectively.

The actual convolutions were carried out by a LISP machine constructed at the MIT Artificial Intelligence Laboratory, using additional hardware specially constructed for the purpose.

II Detection and description of zero-crossings

In theory, the elements that are matched between images are (i) zero-crossings whose orientations are not horizontal, and (ii) terminations. From the point of view of the false target problem it is the zero-crossings that cause the real difficulties, and our program uses only zero-crossings.

Because we can ignore horizontally oriented segments, the detection of zero-crossings can be accomplished by scanning the image along horizontal lines, looking for either two horizontally adjacent pixels containing convolution values of opposite sign, or three horizontally adjacent pixels, the middle one of which is zero, and the other two containing convolution values of opposite sign.

In addition to their location, we need the signs of the zero-crossings, and a rough estimate of their local orientation. In the present implementation, the orientation at a point on a zero-crossing segment is computed as follows. Assign some arbitrary but consistent axis in the image. Find the line through the point in question which minimizes the total separation, from the line itself, of the points of the zero-crossing in the neighbourhood of the point.

III Matching

Matching between the left and right images takes place between zero-crossings obtained from the same sized mask, having the same sign, and having roughly the same orientation. The matching process begins with a rough disparity value for the region, whose derivation we describe below, and it implements the second of the matching algorithms described by Marr & Poggio (1979).

Given a zero-crossing in the left image, the search for a counterpart in the right image is centered on the corresponding location there, shifted by the *a priori* disparity value. The area to be searched is divided into three pools, two larger convergent and divergent regions, and a smaller one lying centrally between them. Together these pools span a disparity range equal to $2w$, where w is the (corrected) width of the central excitatory region of the corresponding convolution mask.

If one zero-crossing of the appropriate sign and orientation (within 30°) is found within a pool, the location of that crossing is transmitted to the matcher. If two candidate zero-crossings are found within one pool, (an unlikely event), the matcher is notified and no attempt is made to assign a match for the point in question. If the matcher finds a single crossing in only one of the three pools, that match is accepted, and the disparity associated with the match is recorded in a buffer. If two or three of the pools contain a candidate match, the algorithm records that information for future disambiguation.

Once all possible unambiguous matches have been made, the algorithm attempts to locally disambiguate points with double or triple matches. This is done by searching a neighbourhood about the point in question and recording the disparity sign of the matches within that neighbourhood. If the ambiguous point has a potential match of the same disparity sign as the dominant type within the neighbourhood, then that is chosen as the match. (This is the "pulling" effect of Julesz & Chang 1976). Otherwise, no match is assigned to the point.

Two points remain to be specified. The first is how the *a priori* disparity value mentioned at the beginning of this section is obtained. For the largest mask, this rough shift value is initially taken to be zero; in other words, the images are assumed to be aligned. For the smaller masks, the initial rough shift value is found by averaging the values obtained by the previous mask, for the k closest neighbours to the point. One could also use a histogram of local values to obtain the rough shift value. However, for areas which contain pieces of two distinct surfaces, separated by a sharp discontinuity, the histogram will be forced to choose a value corresponding to disparities belonging to only one of the two surfaces. If the discontinuity is large enough, this may cause the portions of the second surface to be shifted out of the range of the next mask, and hence such points will not be assigned valid matches. In the case of an average of the local values, this will not happen. Instead, the average value will lie somewhere between the disparities of the two surfaces. Hence, it is possible that there will be candidate matches for both surfaces which are within the range of the matcher. In the case of areas which contain only pieces of the same surface, there is virtually no difference between the use of a histogram and the use of a local average.

Finally, there is the possibility that the region under consideration is not within the $2w$ disparity range. This situation is detected and handled by tesselating the image, and within each tesselation square, performing the following operation. Given a tentative shift value, an attempted matching is performed for all the zero-crossings within a particular tesselation. Any crossing for which there is no match is marked as such. If the percentage of unmatched points exceeds a threshold of 0.3 (Marr & Poggio 1979) then the region is declared to be out of range and a new shift value is tried. The process continues until either the region comes into range or all possible shift values are exhausted.

The overall effect of the matching process, as driven from the left image, is to assign disparity values to most of the zero-crossings obtained from the left image. An example of the output appears in the figures. In this array, a zero-crossing at position (x,y) with associated disparity d has been placed in a three-dimensional array with coordinate (x,y,d). For display purposes, the array is shown as viewed from a point some distance away. The heights in the figure correspond to the assigned disparities.

IV Overall structure of the process

The complete algorithm, as we have implemented it, uses three mask sizes. To begin with, the two views of the scene are convolved with the largest mask and are passed to the algorithm. The zero-crossings and their orientation are computed. An initial horizontal and vertical registration of zero is assumed for the entire image and the matching process is performed under this assumption. Any points with either ambiguous matchings or with no match are marked as such.

Any tesselation square whose percentage of unmatched points exceeds a certain threshold assigns a new registration value for that square and the matching is performed again. This continues until either all tesselations are matched or all possible registrations have been attempted. The ambiguously matched points are then considered, and the pulling effect is applied to them. If ambiguity still remains after this process, the point is not assigned a match.

The algorithm now passes to the next smaller mask and repeats the process of obtaining the zero-crossings and their orientation. For the matching process, the initial registration for each tesselation is simply the average of the disparities assigned in the neighbourhood by the previous matching process. The rest of the process proceeds as before.

The final output is thus a sparse disparity map, with disparities assigned along most portions of the zero-crossing contours obtained from the smallest masks.

Examples and Assessment of Performance

The implementation of the stereo theory was tested on a number of images. This section contains examples of the various stages of the algorithm, run on several images, and an assessment of the performance of the implementation.

A good tool for testing the performance of the implementation, and hence of the theory, is the random dot stereogram. For example, a random dot stereogram consisting of a plane separated in depth from a second plane contains well demarked disparity values. Thus, the performance of the implementation can be easily assessed.

The first pattern had a dot density of 50%. Each dot was a square four pixels on a side. This corresponds to a dot of approximately two minutes of visual arc. The total pattern was 256 pixels on a side. The actual region on which matching took place was 150 pixels on a side. The central plane of the figure was shifted 12 pixels in one image relative to the other. In the final disparity map assigned after the matching of the smallest channel, approximately 0.6% of the points were incorrectly matched.

A similar test was run on a pattern with a dot density of 10%, and one point (0.1%) was incorrectly matched. For a pattern with a dot density of 5%, every point was correctly matched.

Discussion

The figures exhibit the analysis for two images, a 50% random-dot stereogram and a natural image, in this case of a sculpture by Henry Moore. Although one cannot make precise measurements for natural situations, the program appears to be performing on them at least as well as on the more difficult stereograms.

References

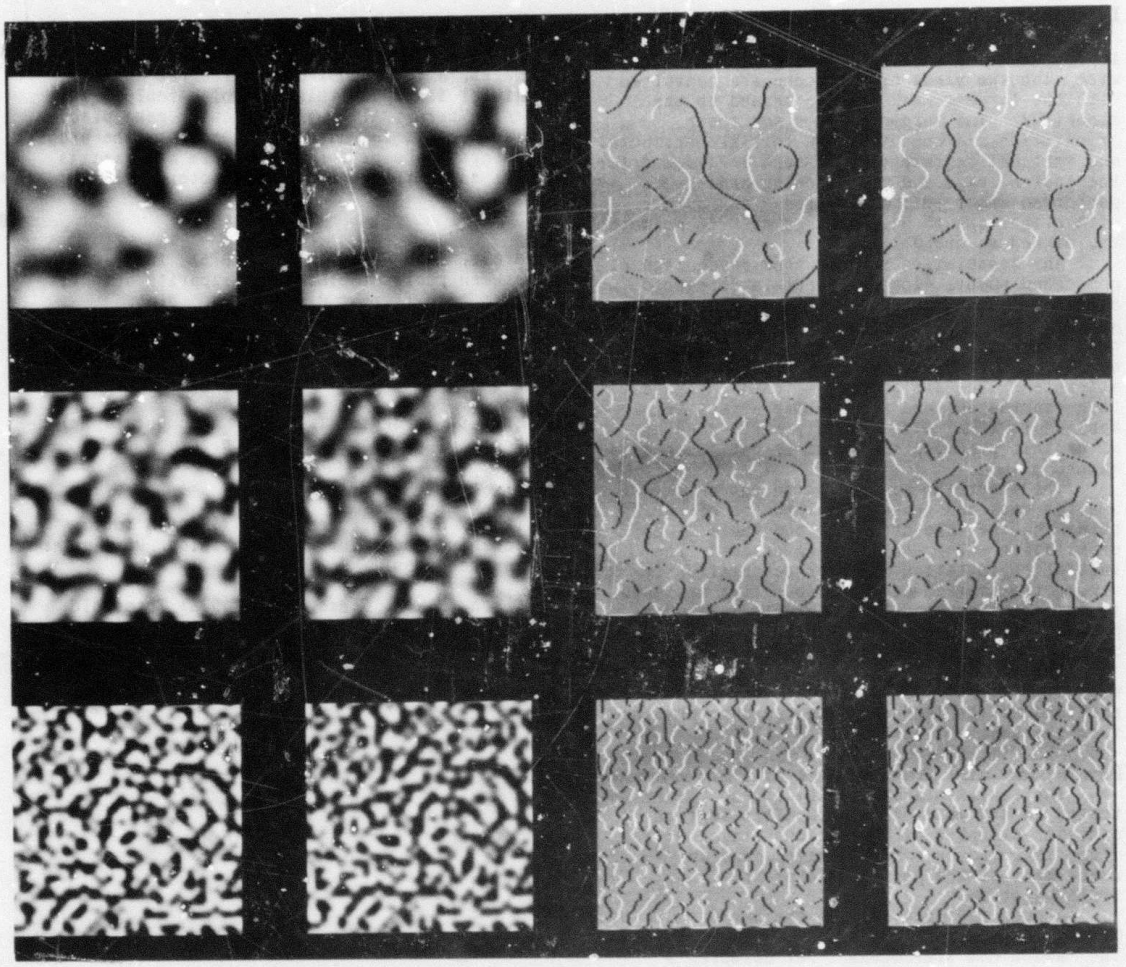
Julesz, B. & Chang, J.J. (1976) Interaction between pools of binocular disparity detectors tuned to different disparities. *Biol. Cybernetics 22*, 107-120.

Marr, D. & Hildreth, E. (1979) Theory of edge detection. *Proc. R. Soc. Lond. B* (in the press).

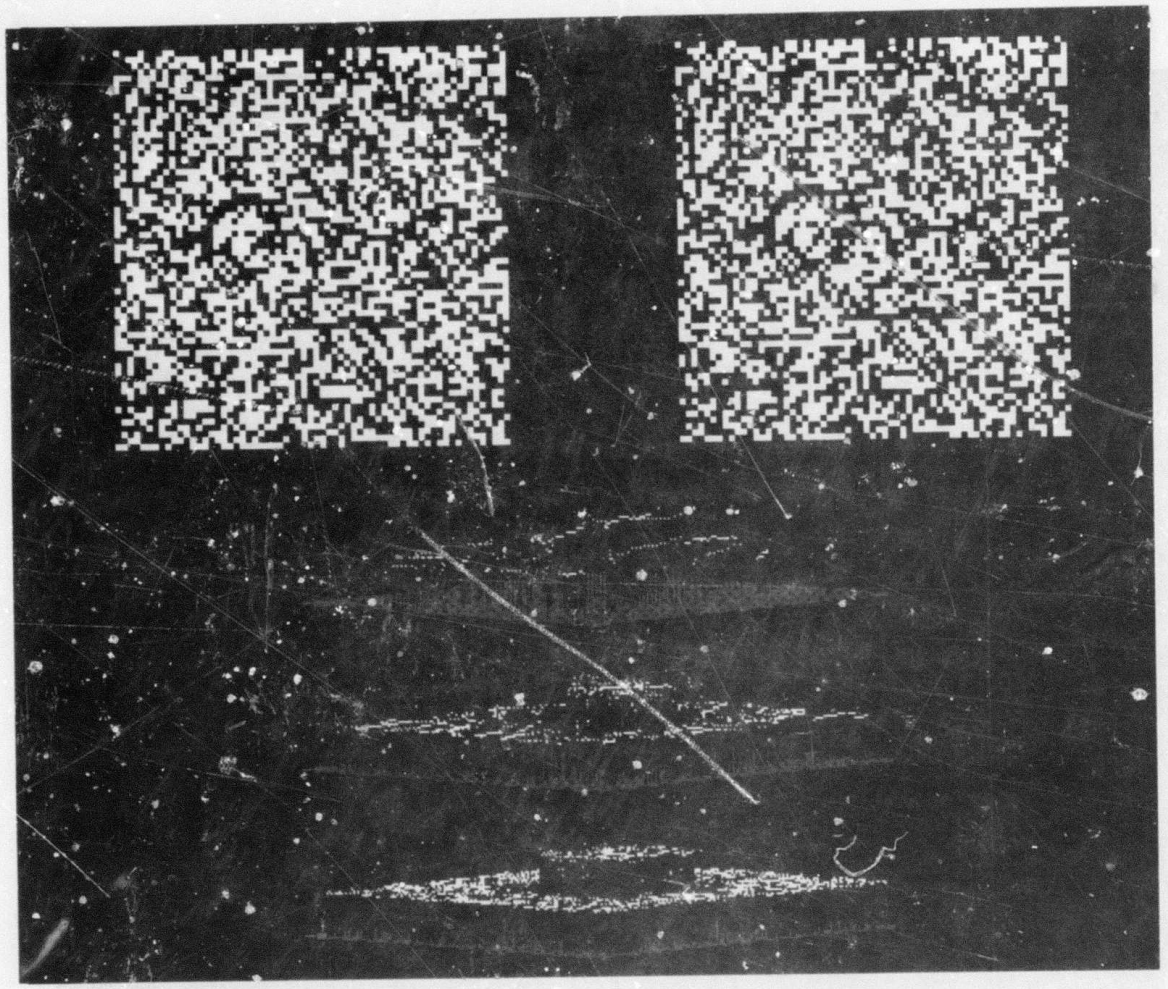
Marr, D. & Poggio, T. (1979) A computational theory of human stereo vision. *Proc. R. Soc. Lond. B* (in the press). Also available as M.I.T. A.I. Lab. Memo 451 (1977).

Wilson, H.R. & Bergen, J.R. (1979) A four mechanism model for spatial vision. *Vision Res.* (in the press).

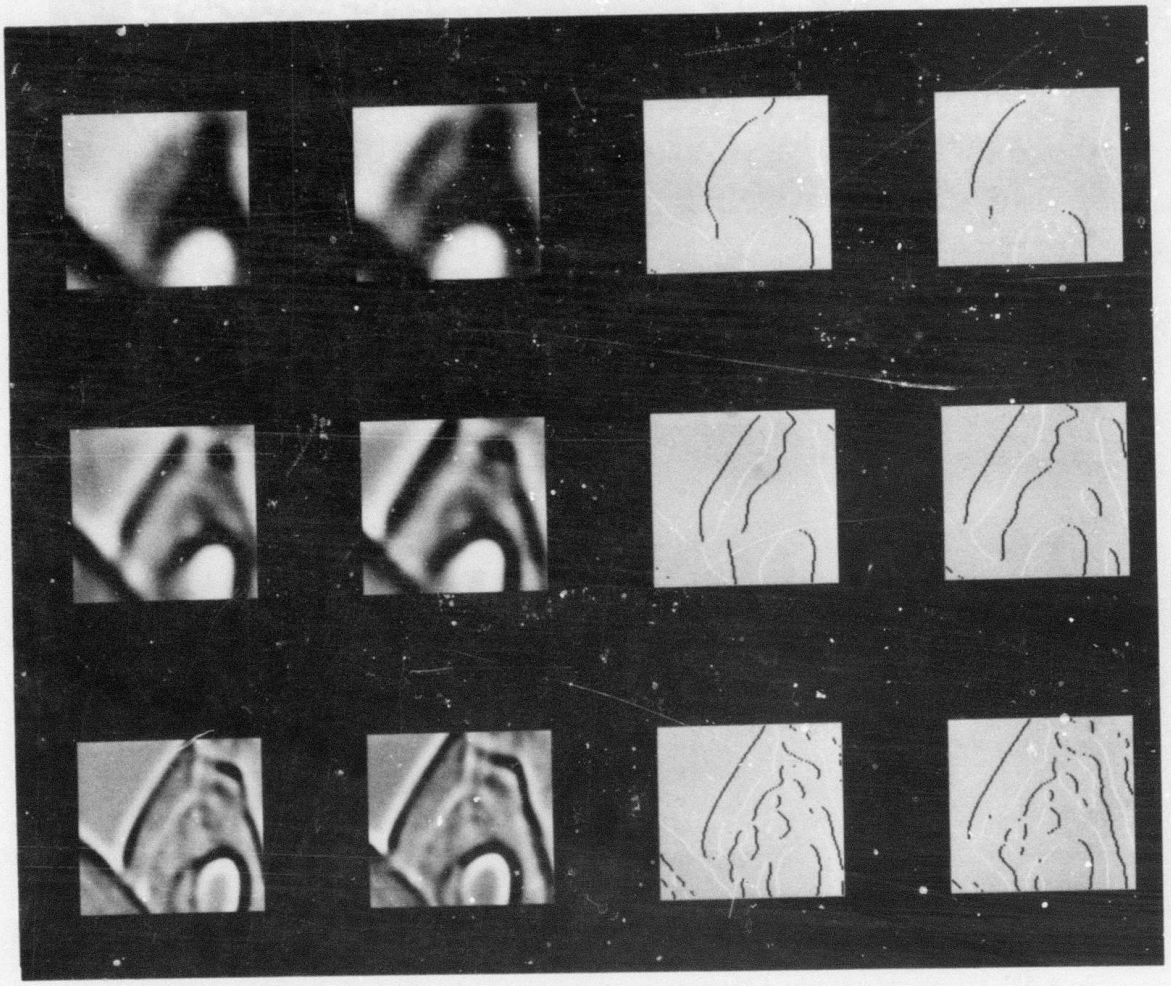
Wilson, H.R. & Giese, S.C. (1978) Threshold visibility of frequency gradient patterns. *Vision Res. 17*, 1177-1190.



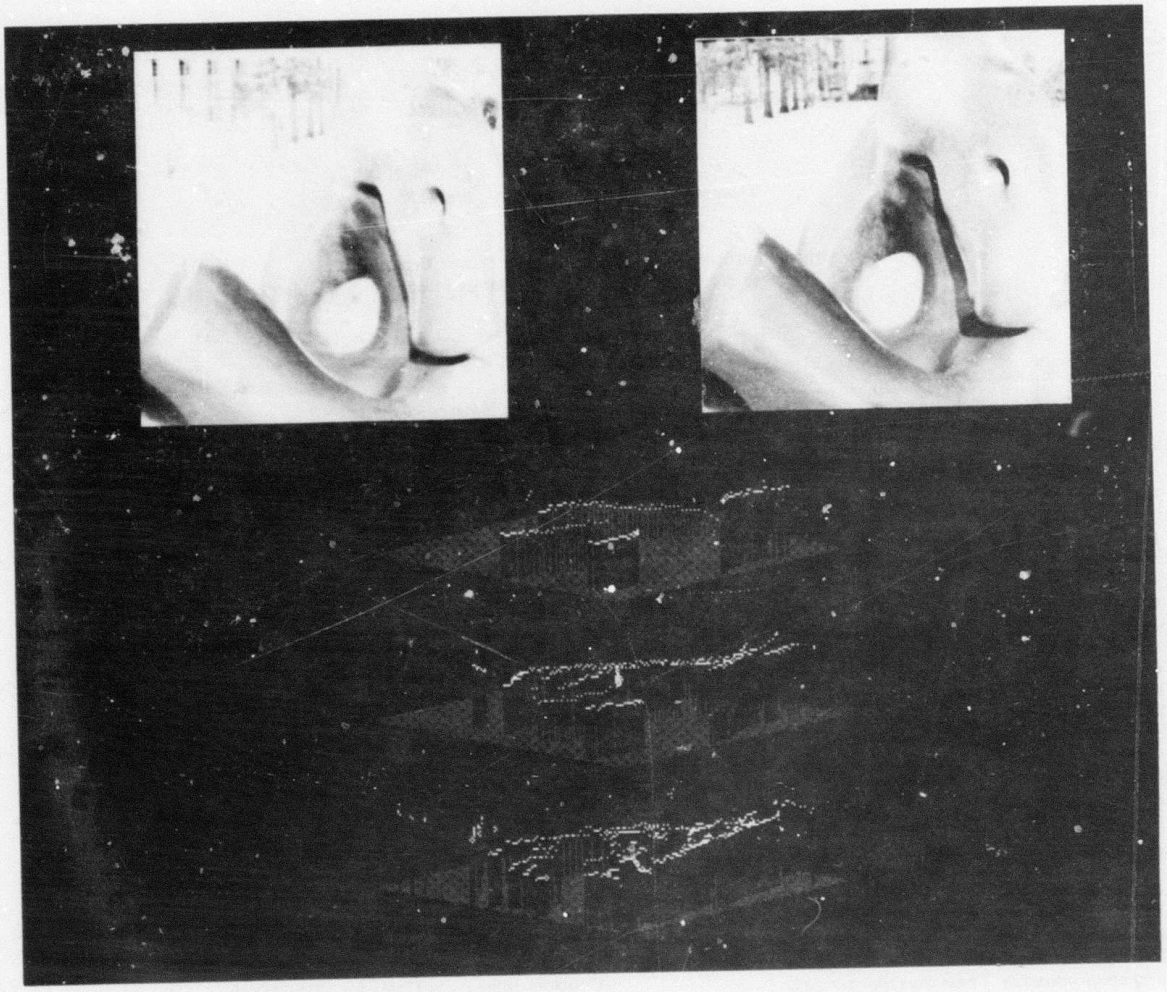
1. A 50% random dot pattern. The two left columns indicate the convolutions of the left and right images with masks of size w = 35, 17 and 9 respectively from top to bottom. The two right columns indicate the zero-crossings obtained from the convolutions in the left most columns.



2. A 50% random dot pattern. The left and right images are shown at the top. The three lower figures indicate an orthographic view of the disparity maps obtained by matching the zero-crossing descriptions of figure 1. A point in the image with coordinates (x, y) and an assigned disparity value of d is portrayed in this three dimensional system as the point (x, y, d). Here, the height of the bright points above the plane indicate the disparity values.



3. A natural image (a Henry Moore sculpture). The two left columns indicate the convolutions of the left and right images. The two right columns indicate zero-crossings obtained from the convolutions.



4. A natural image (a Henry Moore sculpture). The left and right images are shown at the top. The lower three figures indicate the disparity maps obtained from the three different channel sizes.

GEOMETRIC REASONING IN ACRONYM

Thomas O. Binford and Rodney A. Brooks

Artificial Intelligence Laboratory, Computer Science Department
Stanford University, Stanford, California 94305

Abstract

ACRONYM demonstrates mechanisms for generic interpretation of images in a way that is generalizable. It includes a high level modeling language for natural communication with the user in terms of object models and observation models. It uses a rule-based geometric reasoning system to predict symbolic appearances of objects. This geometric reasoning capability enables ACRONYM to integrate knowledge and data at different levels. ACRONYM constructs a Picture Graph using surface descriptors [Arnold] and ribbon descriptors [Brooks] obtained from edge descriptors [Nevatia and Babu]. ACRONYM has a powerful syntactic matcher. This report describes geometric reasoning with generalized cones to build the Observability Graph for aircraft at an airport. Increments to the modeling system were necessary. Initial efforts in formalizing the interpretation process are described. In a separate paper, Brooks describes a rule-based region descriptor which demonstrates high level guidance of a low level description process. ACRONYM is in the midst of its first test on real data, aircraft at a passenger terminal.

Introduction

ACRONYM is intended to demonstrate mechanisms which provide key capabilities for interpreting scenes. It is intended to function within a scenario of cartography and photointerpretation. These capabilities include:

1. Interpretation should be generic with respect to objects and generic with respect to viewing conditions. Typical scenes include aircraft, vehicles and buildings. It does not seem reasonable to separately enumerate and identify all configurations of buildings from all viewpoints. Even if that were possible, it would leave the human designer with the effort of defining similarity classes, e.g. jet passenger aircraft, smokestacks, stairs, etc. Defining similarity classes can be automated. Traditional graphics approaches might be adequate for single object models from single viewpoints but not for generic interpretation. Our approach for interpretation of generic object classes is to use object models constructed from generic parts, which are generalized cones, and to use symbolic generic predictions of elements from these generic parts.

2. Users should be able to specify tasks in a natural and simple way. Geometric models are natural for both the user and the vision system. Ultimately, users will be able to instruct systems in natural language. The representation hierarchy of ACRONYM could serve as a bridge between natural language and standard programming languages.

3. The system should integrate information which comes in different levels and different forms. A photointerpreter solves a puzzle by piecing together selected and multiple clues from current images (image level), background information (object level) and previous images. In doing so, he relies heavily on spatial interpretation from stereo imaging and shadows (surface level), and spatial knowledge about structures (volume level). Integrating multiple cues within a single task raises technical requirements for representation.

4. Systems for very different tasks should be constructed from a large core of common modules and a small set of task-specific modules. For a single system to map this wide range of task elements onto a common set of modules, it is convenient that the modules represent a natural decomposition of the problem into physically meaningful elements, for example, those we use in our own description of the problem. Our basis for generalizability is the use of a tightly structured hierarchy of geometric representations. A photointerpreter performs a wide range of tasks which have widely different collateral information at the contextual level; tasks deal with widely varying objects; they vary greatly at the image level because of varied viewpoint, illumination, sensors, weather, and obscuration and camouflage.

Consider some typical tasks for ACRONYM:
1. A photointerpreter monitors an airfield. The system identifies and counts aircraft at frequent intervals to monitor air traffic.
2. An interpreter monitors a building complex for changes. The system uses stereo, a model of the complex, and identification to distinguish insignificant from significant changes. The system might not notify the interpreter about changes from snow or rain, or moving a vehicle, but notify him about building additions.
3. An interpreter monitors vehicles in staging areas. The system identifies vehicles and monitors traffic to and from the area.

Figure 1 shows the data structures and data flow, and program modules of the ACRONYM system. Data structures are enclosed in the inner box, while program modules surround the inner box.

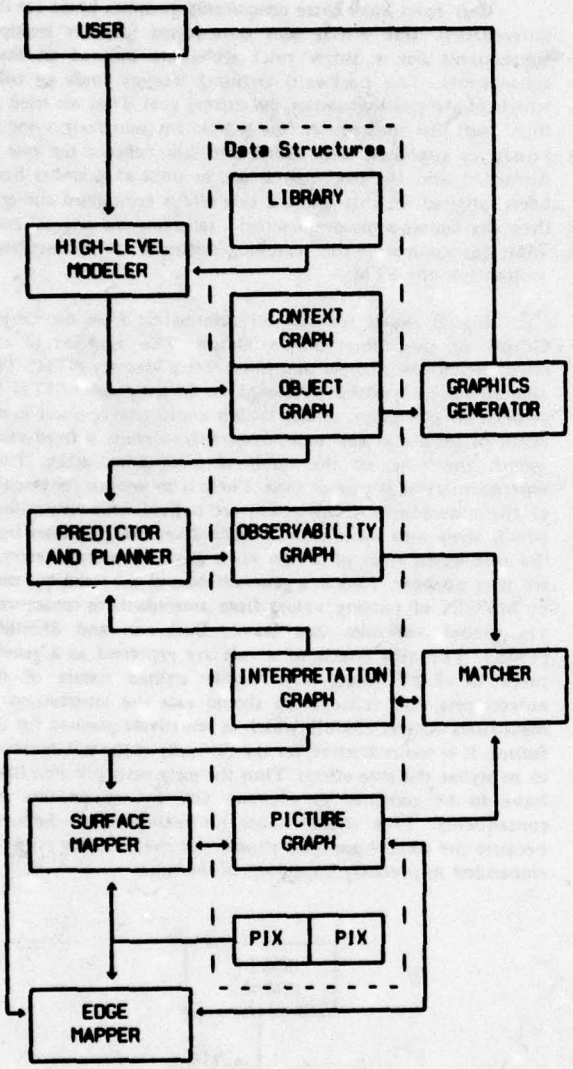


Fig.1. The ACRONYM system.

Geometric Modeling

In previous papers (Brooks, Greiner and Binford [1978a, 1978b]) we have given detailed explanations of the geometric models used by the ACRONYM system. Recently we have made two major generalizations to the representations we use. Geometric modeling supports a broader subclass of generalized cones, and the subpart and spatial relation hierarchies have been separated into distinct and independent structures.

The subclass of generalized cones we now allow can be summarized as follows: 1. The cross sections must have a boundary which can be decomposed into straight line segments and circular arcs. 2. The cross section can be kept at any constant angle while being swept along the spine. Formerly, cross sections were normal to the spine. This generalization enables representing a larger class of objects and is natural for aircraft wings, for example. 3. The sweeping rule must be piecewise linear, and continuous. The spine must be continuous and made up of straight line segments. Circular arcs can also be used as segments of the spine, so long as the sweeping rule is constant, the cross section has a piecewise linear boundary and is kept normal to the spine. The generalization here over our previous system is in allowing piecewise linear sweeping rules, instead of the former linear sweeping rules, and allowing a segmented spine. The class of shapes which is now representable is no bigger than before, as these more complex cones could be built as structured objects. The advantages are in terms of now being able to refer to these shapes as primitive volume elements, where the intent of the spatial relations between object subparts no longer need be deduced.

The other change we have made to the representation was more major. A previous modeling system based on generalized cones Miyamoto and Binford [1975] used a level of detail or subpart hierarchy. It tied the spatial relation hierarchy to this same tree structure. The original version of the ACRONYM modeling system did likewise. Often however, there is no real justification for this correspondence and models must be strangely contorted to accommodate it. As a simple example, consider that in modeling an aircraft it is natural to affix the two wings to the fuselage, so that their position in space is described in terms of the position and orientation of the fuselage. However it is not natural to describe an aircraft at some level of description as merely a fuselage, and include wings only at a refined level. In fact often the wings will appear the same size as the fuselage. By separating the two hierarchies we are able to retain the spatial relation organization described above, while having a top level decomposition of the aircraft into the fuselage and the two wings.

Work has begun on a geometric editor for constructing object models. Initially it will provide some of the capabilities of GEOMED [Baumgart] for convenient interaction, combined with track ball and other analog input devices. Beyond these initial stages, we plan to include geometric reasoning capabilities discussed below to provide the ability to understand implied relations and to draw inferences from examples.

Reasoning about Geometry

The Predictor and Planner is a critical module of the ACRONYM system. ACRONYM is the first vision system to incorporate a general reasoning system. It is necessary because we wish to make symbolic predictions of the appearance of generic objects, from generic viewpoints. We have previously (Brooks, Greiner and Binford [1978b]) discussed the structure of the Observability Graph. Its automatic production is a difficult task which requires reasoning about a large body of diverse knowledge. We have chosen to use a rule-based system to facilitate experimentation, and to provide additivity of new knowledge. We will first describe the control structure we have implemented for our rules, then discuss some of the techniques used in our early sets of rules to produce the Observability Graph.

The basic control strategy we have chosen is to have our rules consequent driven (i.e. goal-directed, or backward chained). The rules can be interpreted to mean that the consequents should be asserted if the conjunction of the antecedents can be proved. It is thus easy to write rules which traverse either the affixment structure, or subpart structure in a depth first manner. This is the search order which provides the natural way to decompose an object - i.e. process the subparts first, then determine the relationships between those subparts to describe the object. The same is true of affixment structures. The depth first nature of the Object Graph traversal means that inadequacies in some given area of the Object Graph will be tend to be noticed around the same time and thus any questions to the user will tend to be naturally grouped into coherent topics of interest (see Davis [1976]). This simple strategy is not quite sufficient for the task at hand, however. In our current system it is possible that once invoked and "fired" a rule may take control for a while, doing some forward reasoning, and possibly setting up new goals which are attempted before control is handed back to the original invoking mechanism. However, depth-first is an unnatural way for strategies for perception. The subpart hierarchy provides an approximate ordering for perceptual search. For example, to find an aircraft, find its fuselage and wings first, then find its stabilizers and engine pods.

Davis, Buchanan and Shortliffe [1975] use consequent-driven rules in the MYCIN medical diagnosis system. Our first implementation of the rule-based predictor and planner closely followed the MYCIN model (Brooks, Greiner and Binford [1978b]). However, we are working with more complex, structured data, and have gradually moved to a rather different system.

MYCIN-like systems use a single, uniformly structured data representation to hold both the original facts and the assertions made during the reasoning process. Rules both read and write into this single collection of object-attribute-value triples. Our current system has three data sites. The Object/Context Graph (including extra interaction with the user) is a read-only site from which originate most of the facts used in a deduction. The Observability Graph is an essentially write-only site. Thirdly, there is an assertion space (or short term memory - STM). This consists of triples of a descriptor, a context and a value and is both a read and write site. Assertions in the short term memory can be thought of as defining the value of a function (the descriptor) on some element of its domain (the context). The descriptor is an S-expression - currently these are used purely syntactically via tests for equality, but later semantic information may be explicitly attached; the context is a structured fragment of the Object graph. Many descriptors are treated as predicates, by the rules, and so only have values true or false. Production of the Observability Graph is the reason for existence of the Predictor and Planner. However internally, the Observability Graph is produced only as a side effect of manipulations of assertions in STM. The reasoning system can be viewed as trying to prove some theorem in the assertion space, by using rules of deduction (described below). For instance the top level goal for producing the Observability Graph for a model called AIRPORT would be to achieve the triple <(OBSERV GRAPH), AIRPORT, T> in STM. In finding a proof of the desired theorem, these rules are invoked, and their side effects build a correct Observability Graph. Thus every valid proof of a theorem, defines some structure for the Observability Graph.

Our rules have three components; premises (these are the antecedents), side effects and consequents (possibly multiple consequents for a single rule). Rules are indexed on their consequents. The backward chaining strategy finds all rules which might possibly satisfy the current goal. They are tried in turn until one succeeds. A rule is tried by recursively trying to satisfy its premises, until either one fails (whence the rule is discarded and the next rule tried), or until all premises have been satisfied. In this case the side effects are carried out, and then the consequents are asserted - satisfying the original goal. Thus the control of the reasoning system is via the assertions written into the STM.

Fig. 2 shows the flow of information from the Object Graph to the Observability Graph. The assertion of the consequents places them into Short Term Memory (STM). The antecedents use stylized accesses of the Object graph or STM. In general an antecedent simply applies a relational operator to the result of an access and some other value - either a fixed value within the rule, or the result of some other access. Each antecedent returns true or false. There is an implicit conjunction of the antecedants. Consequents are stylized write instructions, which write into short term memory. They can get values from the side effect stage of a rule via a push down stack reserved for that purpose. This is a generalization of the technique used in MYCIN of passing values from antecedants to consequents via global variables (see Davis, Buchanan and Shortliffe [1975]). The side effects of a rule are expressed as a general piece of LISP code. The highly stylized nature of the antecedants and consequents should ease the introduction of meta-rules (Davis [1976]), which is tentatively planned for the future. It is too restrictive, for the difficulty of the task involved, to so stylize the side effects. Thus the meta rules will most likely have to be confined to examine just the antecendants and consequents. This should cause no real problems however, because the control and goal structure of the reasoning system is embedded in precisely these parts of the rules.

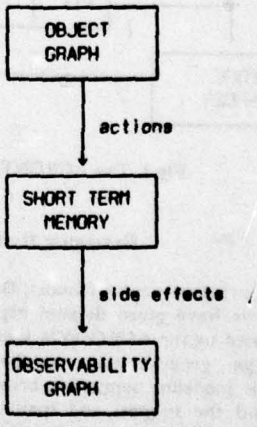


Fig. 5. Producing the Observabilty Graph

In producing the Observability Graph the Predictor and Planner works with observables as defined from characterizing low level feature description operators. It is guided by two classes of quasi-invariants. Object class quasi-invariants broadly determine the node structure of the Observability Graph. Viewing condition quasi-invariants determine the detailed contents of nodes, and the arcs and relations between nodes. Of course this distinction is sometimes blurred. For instance a mixture of the two types of quasi-invariants directs production of the arc which says that runways should intersect a taxiway in the image. The object model says that they should be connected, and a rule which knows that connectivity is invariant under all viewing angles deduces that intersection of the regions corresponding to the nodes will be observable.

At the time of writing we have developed some 42 rules, for use in the predictor and planner. These can be roughly categorized into classes of rules that transfer values from the Object Graph to STM, rules that build Observability Graph node structure, rules for deducing simple shapes, rules for symbolic arithmetic, rules to calculate relative positions of objects, and rules about the camera model. Notice that this categorization is in terms of both the actions, and the side effects, but that the control mechanisms see the rules only in terms of their actions.

The rules which transfer values from the Object graph to STM perform two tasks. Firstly they can greatly improve efficiency and rule readability, as accessing data in the Object graph often requires following a long list of pointers. Accessing the STM is done by an associative hashing mechanism. Thus having rules refer to STM results in both efficient retrieval (with a single complex original retrieval from the Object graph), and a clear and easy syntax within the rule. Many of these simple lookup rules also carry out simple summarizing operations. There are often many ways to represent the same geometric shape in the object graph. This multiplicity is motivated by a desire to provide a user with a natural and intuitive modeling system. Some of the lookup rules map these representations into named classes, which other rules know how to handle. For instance both a square cross section and a rectangular cross section with equal width and height might get mapped into a single class SQUARE in STM (and other classes too, as there are no uniqueness constraints). The classes are represented in STM as descriptors, and something is a member of the class only if it appears as the context of a triple with that descriptor, with value T. The intent of the classes is to summarize the situations of applicability for other rules.

Observability Graph nodes can either describe observable shapes, or can be recursively complete Observability Graphs. For instance for an airport Observability Graph there will be nodes which describe the shape of runways and taxiways, and a node which is the Observability Graph for aircraft. Arcs between these nodes will describe their spatial relations, including such facts as runways are connected to taxiways, and aircraft can be found on runways and taxiways. The class of rules being considered here decides what type of nodes should be generated, and their premises cause goals to be set up to calculate all the relevant details to fill in the node descriptions. Since nodes can be quantified in a variety of ways (e.g. there are from one to three runways in an airport) there are rules to handle these quantifications.

So far the rules we have written for shape description, deal only with very simple three dimensional generalized cones, flat straight ribbons (such as runways) and right circular cylinders. Rules for a much larger class of shapes will be easily written now that we have the developed a framework, without any new conceptual difficulties. Eventually as we tackle very complex shapes, we will have to develop some new tools. Currently the rules take into account any camera model which is known (there are different rules for different levels of knowledge) and predict the shape of a ribbon in the image. To do this requires setting up subgoals to deduce the three dimensional location of the shape in the model, and often arithmetic subgoals. The latter occurs for instance in calculating the apparent width of a ribbon arising from a right circular cylinder. The radius of the model cylinder has to be multiplied by two. This happens in the side effects, where the node in the Observability Graph is being constructed. The problem is that the radius may not be known exactly, but rather only as some description - e.g. a range descriptor, or perhaps a histogram of distribution of expected radii. The side effects of the particular rule which deduces the shape of the right circular cylinders set up a goal of multiplying the radius by 2. The backward chaining mechanism is invoked recursively on this subgoal. There are rules which know how to multiply different representations of quantities. Currently there is a rule which can multiply simple numbers (by invoking the LISP multiply function in its side effects) and a rule which can multiply an interval by a number. The premises of these rules control which one will actually get invoked. The previous example here, is a good example of the additivity of knowledge that the rule system gives. As we include more representations for quantities, we will not need to go and change all the rules which deal with quantities, such as the one mentioned above, but merely add new rules which can carry out the primitive operations, such as multiplication, which we wish to carry out upon quantities.

One way to reason about spatial relations is via matrices of numbers, and matrix arithmetic. The graphics generator module of ACRONYM takes this approach. It is successful in carrying out the tasks desired of it because it has embedded in it general numerical formulas which hold over the whole range of spatial orientations. In the predictor and planner however we are dealing with symbolic descriptions af shapes. Often it is possible to make use of special orientations, and make deductions which hold only for special cases, rather than the general case. Thus there are rules which deduce spatial relations between special named cases, such as the ground plane, a vertical plane, the class of horizontal planes, etc. By carrying out spatial manipulations at this level, explicit knowledge about orientations and positions can be preserved across transforms. A strictly numerical approach would result in this explicit knowledge being lost in an array of numbers, or at best, it might have to be re-extracted by routines which could recognize special cases of numeric arrays.

Interpretation

The interpreter used thus far is a state of the art graph matcher which was described in previous reports. The interpreter matches the Observability Graph to the Picture Graph. Semantics of the problem domain are encoded in the Observability Graph. Ordering of the matching process for efficiency is accomplished with the Observability Graph, also.

There are several ways that efficient matching is carried out. First, the matching process determined by the Observability Graph is tailored to individual problems. An Observability Graph is made for the particular objects and viewing conditions of each problem. There is no fixed set of operations which are performed whether relevant or not. Second, the Observability Graph encodes a coarse to fine matching. For aircraft, this means that first wings and fuselage are matched, then smaller detail like engines and horizontal stabilizers. These levels of matching correspond to levels of detail in the Observability Graph. The structure of the Observability Graph reflects levels of detail in the Object Graph, the main parts of which are fuselage and wings. However, Observability Graph structure depends on knowledge about the process of observation. For example, the smooth intensity gradient along the curved edge of the fuselage may cause problems for most edge operators. In SAR images, wings and fuselage may not show up. The matching process implements this coarse to fine matching structure. We plan to soon implement an improvement to explicitly use the fuselage locations when searching for the wings, and subsequently use these locations to constrain the search space for the stabilizers and engine pods. We intend to incorporate efficient scheduling in a system of rules.

We are working to formalize the interpretation. In previous work, matching has been represented as graph embedding [Barrow]. That is, consider a graph description of an expected image of the object, G; then, matching seeks an isomorphism of G with a subgraph of the feature graph of the picture, what we call the Picture Graph. This representation leads to efficient algorithms for matching, but there are several problems with this approach. The first is that feature descriptor algorithms have serious limitations; e.g.edge finders miss edges. The second is more fundamental: objects may be camouflaged, or have snow or rain on their surfaces. Even more fundamental problems arise when we seek generic object classes.

One approach to dealing with anticipated differences is to define a distance metric. This approach adds very little capability to the initial matching technique, although it adds considerable complication in conceptualization and computation. Distance thresholds large enough to accomodate markings and configuration differences offer little discrimination.

One approach to formalization will be presented in a forthcoming memo. Another approach to matching is generalized invariance, described here by analogy with generalized translational invariance. Generalized translational invariance is the principle underlying generalized cones. Consider two cross sections. The guiding principle was not equivalence of two cross sections within a distance measure under a suitable translation, but instead congruence of the two cross sections under a translation and normalizing transformation.

We represent the matching process as:

$$T \circ O = I \circ P,$$

that is, an isomorphism of a subgraph of volumes obtained from the Object Graph with a subgraph of volumes lifted from the Picture Graph. Let us explain. O is the Object Graph and P is the Picture Graph. O may be a generic or specific object, that is it might represent the class of jet passenger aircraft or a 747. T is a transformation from the 3d Object Graph to a closely related 3d model. I is a 3d interpretation operation which maps from 2d image features, ribbons or edges, to 3d volumes, generalized cones. I also maps from surface features obtained from stereo to 3d volumes.

For example, consider the class of coffee cups. They include styrofoam conical cups and handmade pottery specimens. They differ greatly in shape, volume and trim; some have handles, others not; some are cylinders, others distinctly conical; A generic description for this class is a functional description. from the beginning of our work with generalized cones, our guiding principle for dealing with objects has been form = function. Freiling demonstrated an interesting program based on this principle [unpublished]. Concretely, that means that if we can describe the function, we can describe the relevant form, and use shape to represent classes with enormous variation. In this example, a coffee cup is a container for fluids which is to be held in a human hand. This gives a generic specification of the class of cups: it is a container, hence enclosed except vertically. From a little more knowledge we infer it is open at the top and flat on the bottom, and something about the material it is made from. We infer its approximate volume and its approximate diameter from human measurements of thirst and hand size. From a knowledge of fabrication methods we infer its usual circular cross section, and flat top. In this discussion, O is the class of coffee cups, and T is the transformation from O to the generic description we arrived at, a hollow circular cone with diameter approximately specified, flat ends, closed bottom, open top, vertical orientation, made of one of a few materials. We believe that this inference capability provides powerful generality at the expense of a moderate knowledge base. ACRONYM does not have this capability now.

A measure is required to compare alternative interpretations. There are many examples in which humans ignore feasible but unreasonable interpretations of scenes. We assume that a set of preferred interpretations are used and that they are in some sense canonical. We assume that the comparison does not involve small differences in a distance measure, but major differences in structural descriptions [Nevatia 1974, Nevatia and Binford].

The interpretation map I includes operators which predict obscuration, shadow and illumination, and the effects of perceptual operators including missing edges, limited resolution, and noise. These are all quantifiable and their consistency in the final interpretation can be tested. The advantage of this formulation over a definition of a match as a thresholded distance function is that weights in the evaluation depend greatly on object semantics for each element of the match. Surface markings do not relate to characterizing underlying volumes, hence they do not enter into the match evaluation. Interpretations will thus be of the form: this is a 747 with the nose section obscured by a passenger ramp. The wing is obscured partially by the fuselage and there are markings. Interpretation operators may be top-down or bottom-up.

Experiments

In this section we briefly review some tasks which have already been carried out by ACRONYM. The system has not yet found an instance of an object in a digitized image given only a high level description of the object class and low level descriptors of the image. However, that achievement appears near. All the necessary mechanisms for such a test are operational in at least prototype form, and have been tested individually or as subsystems smaller than the total ACRONYM system. As more rules are written for tha predictor and planner we expect to soon be able to run a complete test.

The High-level Modeler has been used to construct a large number of models. Examples of output from the Graphics Generator have been presented in previous workshop papers. The Modeler and Graphics controller are completely interactive, enabling the user to "fly" around airports or more generally to orient a partially built model quickly and easily as one would orient a physical model by hand, to examine any desired detail.

The Predictor and Planner has been run on a specific model of an aircraft and a generic model of an airport. It produced the complete node structure of the Observability Graphs in both cases, and included quantifiers in the airport case, to describe the numbers of allowable numbers of runways and taxiways in a valid airport instance. In separate tests the Predictor and Planner has deduced the expected shapes of simple generalized cones.

The Matcher has been tested on a hand coded Observability Graph, matching against a hand coded Picture Graph. This test was used during the debugging phase, and was designed to test all modes of operation of the Matcher. Thus the particular Observability Graph used is more complex than can be reasonably be expected to be generated by the Predictor and Planner, at least until we have had considerably more experience.

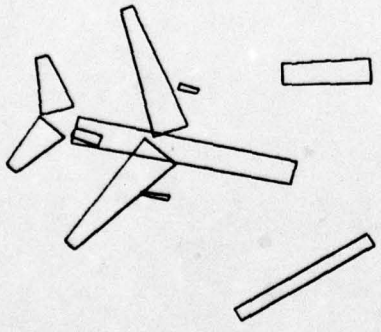


Fig. 3. An L1011.

Recent work on the edge mapping module has produced the results reported in greater detail elsewhere in these proceedings (Brooks [1979]). Figure 3 shows the results of simulating the matcher by hand to control the edge mapper. The ribbons produced are a result of three invocations of the edge mapper; one invocation to find candidate ribbons for wings and fuselage (it found the two wings, the fuselage, a passenger ramp, and a large shed), one to find the rear stabilizers and one to find the engine pods. The last two invocations were directed by the inferences made as a result of the first.

Work is also underway to transport the line finding programs of Nevatia and Babu [1976] to our laboratory. This added self reliance should allow us much more flexibility with our experimental procedures, and also give us the option of forcing some goal-direction down to the line finding level. Such a capability should prove useful when the predictor and planner has an understanding of the process of shadow formation. The line finder will be able to be directed to search with more global information for lines in low contrast areas.

Other work on stereo, sponsored by ARPA is proceeding at our lab. We intend to merge these results into ACRONYM, in the surface mapping module. Depth information will give us surface descriptions to match against, and enable us to use much more of the three dimensional information contained in our models.

References

R.D. Arnold; "*Local Context in Matching Edges for Stereo Vision*"; Proceedings Image Understanding Workshop, Boston, May 1978.

Barrow,H.G., A.P.Ambler, R.M.Burstall; "*Some Techniques for Recognizing Structures in Pictures*"; Int Conf on Frontiers of Pattern Recognition, Honolulu, Hawaii, Jan 1971.

Brooks, Rodney A. [1979], *Goal-Directed Edge Linking and Ribbon Finding*, in these proceedings.

Brooks, Rodney A., Russell Greiner and Thomas O. Binford [1978a], *A Model-Based Vision System*, Proc ARPA Image Understanding Workshop, Cambridge, May., 36-44.

Brooks, Rodney A., Russell Greiner and Thomas O. Binford [1978b], *Progress Report on a Model-Based Vision System*, Proc ARPA Image Understanding Workshop, Carnegie-Mellon, Nov., 145-151.

Davis, Randall [1976], *Applications if Meta Level Knowledge to the Construction, Maintenance and use of Large Knowledge Bases*, Stanford AIM-283.

Davis, Randall, Bruce Buchanan and Edward Shortliffe [1975], *Production Rules as a Representation for a Knowledge-Based Consultation Program*, Stanford AIM-266.

Freiling, M.; unpublished, A I Lab, MIT

Marr, D. and H. K. Nishihara [1976], *Representation and Recognition of the Spatial Organization of Three Dimensional Shapes*, MIT AIM-377.

Miyamoto, Eiichi and Thomas O. Binford [1975], *Display Generated by a Generalized Cone Representation*, Conference on Computer Graphics and Image Processing, May.

Nevatia, R; "*Structured Descriptions of Complex Curved Objects*

for Recognition and Visual Memory" Stanford University Artificial Intelligence Laboratory, Memo AIM-250, CS-464 October 1974.

R.Nevatia and T.O.Binford; "*Description and Recognition of Curved Objects*"; Artificial Intelligence, Vol 8, p 77, Feb 1977.

Nevatia, Ramakant and K. Ramesh Babu [1978], *Linear Feature Extraction*, Proc ARPA Image Understanding Workshop, Carnegie-Mellon, Nov., 73-78.

Thomas, A. J. and T. O. Binford [1974], *Information Processing Analysis of Visual Perception: A Review*, Stanford AIM-227.

DESCRIBING NATURAL TEXTURES*

Ram Nevatia
Keith E. Price
Felicia Vilnrotter

Image Processing Institute
University of Southern California
Los Angeles, California 90007

ABSTRACT

There have been many different approaches to texture description, primarily statistical techniques although there has been some work on structural texture analysis all along. We present here a technique which can be used to easily derive parts of the structural description - the regularity information in particular. Some limits on this method and its use in an overall texture description system are discussed.

INTRODUCTION

Many times, areas of an image are best characterized by their texture rather than purely intensity information. Texture is most easily described as the pattern of the spatial arrangement of different intensities (or colors). The different textures in an image are usually very apparent to a human observer, but automatic description of these patterns has proved to be very complex. We are concerned with a description of the texture which corresponds, in some sense, to a description produced by a person looking at the image.

Many statistical textural measures have been proposed in the past [1-4], therefore one can use some of their results indicating what measures may be useful. Among the statistical measures which have been discussed, and used, are analysis of the discrete Fourier transform to find indications of the structure [4], analysis of generalized gray-level co-occurence matrices [1], and analysis of the edges (or micro-edges) in a subwindow [3]. We are not interested in finding one texture measure which will distinguish between all regions (this is the ultimate, but extremely difficult problem) but in finding a texture measure to use in conjunction with many other features of the region [9].

*This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Wright Patterson Air Force Base under Contract F-33615-76-C-1203, ARPA Order No. 3119.

The work in what can be called structural texture description has been more limited [5-7]. Maleson [5] used simple regions as the basic elements and used relations between regions and shape properties of the region in his analysis. Tamura et al. [6] tried to develop a set of operators which would rate textures on several scales, comparable to their ratings by human subjects. The proposals of Marr [7] for texture analysis based on the primal sketch are similar to some of the analysis which we perform.

ANALYSIS OF TEXTURE

One of the most striking patterns seen in aerial images of a certain scale is the regular street or housing pattern of many cities (see Fig. 1). The appearance of this regularity is its most distinguishing characteristic, and because the pattern is so clear in the image it should be easy to extract. An obvious method to extract this regular pattern is the use of a 2-dimensional discrete Fourier transform. We computed this for various subwindows from the image in Fig. 1 and other images (subwindows are given in Fig. 2). In the Fourier transform results shown in Fig. 3 there is some indication of the regular structure in the urban area windows, but it is not as apparent (visually) as it is in the image. Other attempts to derive much of the structural information from the Fourier transform were only partially successful [4], so we felt other methods should be attempted.

The individual textural elements could be located and analyzed [5], but the simple regions seem to be unreliable when the textural elements are very small, which is the case in the urban areas. Another option is to analyze an edge image to find the structure. The patterns in the original image will cause related patterns to appear in the edge image, and those patterns should be more consistent and easier to analyze than the original image data.

To study textures which are composed of small basic elements, a small window size edge detector must be used. We are interested in the edges between adjacent textural elements and not so much in edges between adjacent textural patterns. The edge operator which we use has been used successfully for other types of

analysis [8]. The operator is applied over a 3 x 3 window and generates an edge magnitude and direction (1 of 8 directions). The direction is defined so that the brighter side is to the right when facing in the direction of the edge. Figure 4 shows the result of applying this operator to each of the subwindows in Fig. 2. The edge data must be further processed before it is in a form useable in texture analysis. Since an edge in the image appears as a broad peak in the edge detector output (the width in this case is two for a perfect step edge), the edges must be thinned. For the experiments here a simple non-maximal suppression was applied in 2 directions (horizontal and vertical), but a more sophisticated suppression which considers the directions of the edge elements could also be applied [8].

The suppressed edge images retain the regularity of the initial image, but now the regularity is in the spacing of edge elements not texture elements. A Fourier transform applied to this binary edge image would indicate the repetitive nature of the binary image, but is obscured by the degeneracies introduced by the binary nature of the input. Generalized gray level co-occurrence computations [1] have been studied for texture analysis, and were intended to indicate sizes of textural elements involved in the pattern. These can be applied more easily to a binary image than a general intensity image to indicate the spacing of edges.

EDGE CO-OCCURRENCE ANALYSIS

Generalized gray level co-occurrence matrix analysis is a basis for much of the statistical texture analysis. Basically, a set of matrices are computed for a portion of the image one for each selected spacing and angle. The entry in the matrix at row I and column J is incremented each time the first image point has the value I and the point at the given spacing and direction has the value J. Usually the image values are partitioned into a small set of values (8 rather than 256), so that it is even possible to compute the initial matrix. Also the computation is applied for many spacings (1,2,3,8, etc.) and several directions (0^o,45^o,90^o,etc.) as shown in Fig. 6. Because of the large number of large matrices that are generated by this method various measures are computed on the matrix values, and the classification is performed using these measures [1]. The common and useful measures do not seem to capture the important feature in the edge images: the regular spacing of edge elements, but this is available in the co-occurrence matrix itself.

When binary edge images are used for co-occurrence analysis, many simplications in the computation can be made. We will use a 1 to indicate an edge at a given point, and 11 to indicate edges occurring at both the first point and the second point which is at some distance and angle from the first. The edge/no edge pair is indicated by 10 and no edge/edge by 01. Finally 00 means no edges at either point. The 10 and 01 combinations mean the same thing in terms of the image and thus are combined. The most important numbers are the 11 totals. The absolute magnitude is not very meaningful since this depends on the total number of edges and on the spacing being used (within a given image there are more opportunities for a co-occurrence edges with a small spacing than a large spacing) in addition to the actual frequency of occurrence of 11's. One good way to normalize the numbers seems to be to use the total of 10, 01, and 11. This gives the proportion of potential edges for co-occurrence that actually co-occur. We computed these values for 4 directions and spacings from 2 to 32 (at 45^o and 135^o a spacing of 2 is plotted at a distance of $2\sqrt{2}$). Some of these results are given in Fig. 7.

There are several ways to compare edges at two points, with different features indicated by the different comparison methods. Using all edges for every direction presents severe problems in the analysis of the output since long lines running in the same direction as the co-occurrence computation will be included along with lines running perpendicular to the direction. (Tamura et al. [6] used this feature to determine linear patterns in their texture experiments.) But, the edge element directions are available and can be used to separate these two different patterns. The first step is to consider only those edge elements perpendicular to the direction of search, that is in the computation of co-occurrences in a horizontal direction only vertical edges are considered. There are an almost unlimited number of variations on this basic restriction which can either be derived from other variations or computed in a manner similar to the simple cases. The variations include: allow some freedom in the edge direction (45^o either way), accept only perfect matches (up and up, down and down), accept only opposites (up and down, not up and up), and allow some freedom in the direction of the last two. The diferent combinations will all produce results with different information, so that several different ones can be computed.

DISCUSSION

None of this analysis would be worthwhile if it did not make the job of describing regular textures any easier. The highly regular patterns of the San Francisco urban area (the top row of Figs. 2-5 and Fig. 7a, 7b) and raffia (the bottom row and Fig. 7c, 7d) produce strong periodic patterns in the plot of the co-occurrence measure. A high value in the graphed measure indicates that edges frequently occur at that particular spacing. This spacing information can be used to determine the size and spacing of the textural elements, and the overall strength of the peak can be used to determine how regular the pattern is.

The spacing of pairs of textural elements is given by the peak to peak spacing using the measure which matches edges only in the exact same direction (as in Fig. 7a,c). The size of individual elements is best given by the measure which allows only edges in the opposite direction (as in Fig. 7b,d). The solid line in the graph indicates the size of dark objects and the dotted line the size bright objects. The size is from the first major peak, the succeeding peaks are caused by the repeated pattern. By comparing the results from the 4 directions, the orientation of the texture can be predicted. Since patterns usually do not line up with one of the 4 directions there will be some contribution to 2 of the directions. When these directions are 45° apart the dominant direction is probably between them (as in San Francisco, Fig. 7a,b). But when they are 90° apart there should be a regular pattern in two directions (as in Raffia, Fig. 7c,d). Thus, from the data we can say that the San Francisco subwindow has a regular pattern of bright and dark regions oriented in one direction, near 45°, with the bright regions being larger (width about 10 pixels) than the dark ones (width about 4). Note that the size of the blocks in the other direction is near the size limit of the co-occurrence computation and also that very few of the edges at the ends of the blocks are detected.

The irregular textural patterns (e.g.the suburban areas of the second row of Fig. 4, and the grass and sand of the third row, first and second windows) do not produce the same clearly periodic patterns of raffia as shown by Fig. 8a,b (for grass and suburban, respectively). But it is possible to derive certain useful features from these results, primarily that of the size of the textural elements. The strong peak near 3 for grass and 4 or 6 for suburban indicates a dominant size for textural elements (in the case of suburban probably 2 different sizes). The graphs indicate that the grass has thin dark and bright textural elements, predominately vertical and to a lesser extent, horizontal. The suburban area has only bright regions somewhat larger. These descriptions still leave open the question of whether the texturall elements are long and thin or small and round. The lack of a substantial peak in the 45° or 135° direction for grass indicates that it is probably long and thin and the small, though readily apparent peak in the graphs for the suburban windows indicates that the regions are probably small and round or more likely,rectangular).

This is not a complete description of the textures, but serves as a good initial description of the patterns. There are still other important features of the textures which are not derived by this method, but could be computed by other techniques. This procedure has been applied on many other irregular and nonregular, patterns with results similar to those for the windows in Fig. 2 [11].

CONCLUSIONS

General texture analysis is a very difficult problem, but this analysis of edge images appears to be an effective method to extract many important structural features from the textural patterns. One major unanswered question is whether or not all of the information derived by the human user can be reliably derived by a program. We are still working on the automatic extraction of this information from the data which is produced by this textural analysis method.

REFERENCES

1. R.M. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," IEEE Trans SMC-3, 1973, pp. 660-621.

2. J. Weszka, A. Rosenfeld, "A Comparative Study of Texture Measures for Terrain Classification," IEEE Trans SMC-6, April, 1976.

3. A. Rosenfeld and Kak, Digital Picture Processing, Academic Press: New York, 1976.

4. R. Bajcsy, "Computer Identification of Visual Surfaces," Computer Graphics and Image Processing-2, Oct. 1973, pp. 118-130.

5. J.T. Maleson, "Understanding Texture in Natural Images," University of Rochester, Ph.D. Thesis, to appear.

6. H. Tamura, S. Mori, T. Yamawaki, "Textural Features Corresponding to Visual Perception," IEEE Trans SMC-8, June, 1978, pp. 460-473.

7. D. Marr, "Early Processing of Visual Information," AI-M-340, Artificial Intelligence Laboratory, MIT, Cambridge, MA. 1975.

8. R. Nevatia, and K. R. Babu, "Linear Feature Extraction and Description," submitted for publication to IJCAI-79.

9. R. Nevatia, and K. Price, "Locating Structures in Aerial Images," Proc. of 4th Intl. Joint Conf. on Pattern Recognition, Kyoto, Japan, Nov. 1978, pp. 686-90.

10. P. Brodatz, Textures, New York, Dover, 1966.

11. R.Nevatia, K.E. Price, and F. Vilnrotter, "Describing Natural Textures," USCIPI Report 660, University of Southern California, Los Angeles, Ca., March 1979.

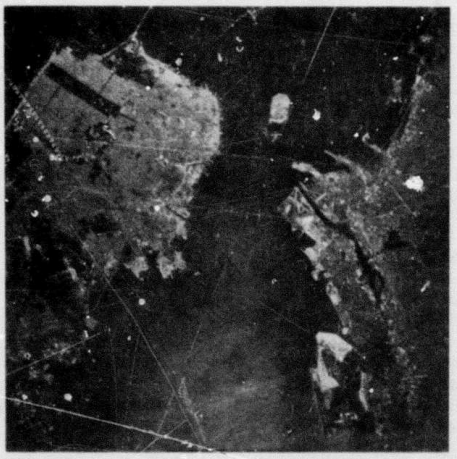


Fig. 1. Aerial View of San Francisco Area.

Fig. 2. 16 Subwindows for Texture Analysis.

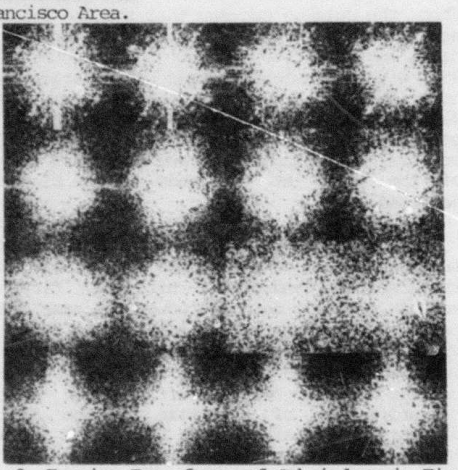


Fig. 3. Fourier Transforms of Subwindows in Fig.2.

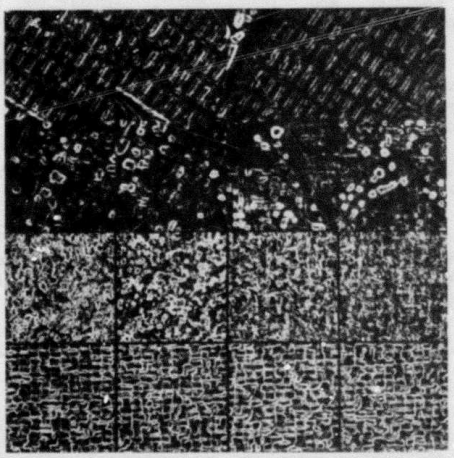


Fig. 4. Edge Magnitude for Subwindows in Fig. 2.

Fig. 5. Non-maximal Suppressed Edges from Fig. 4.

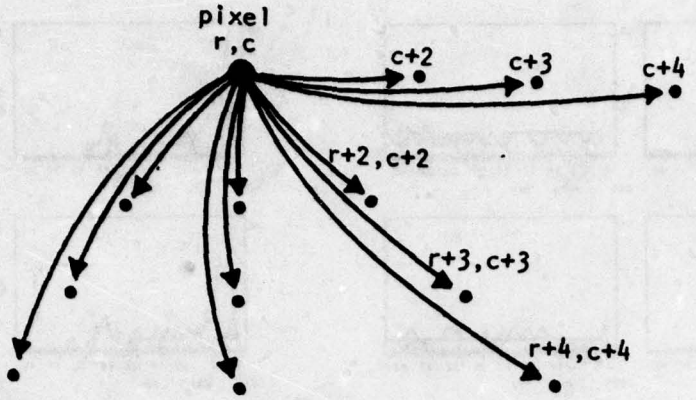
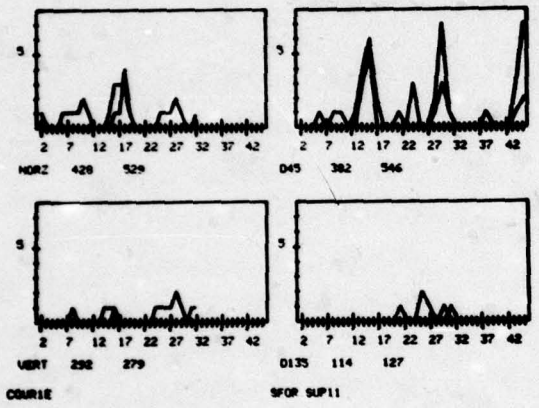
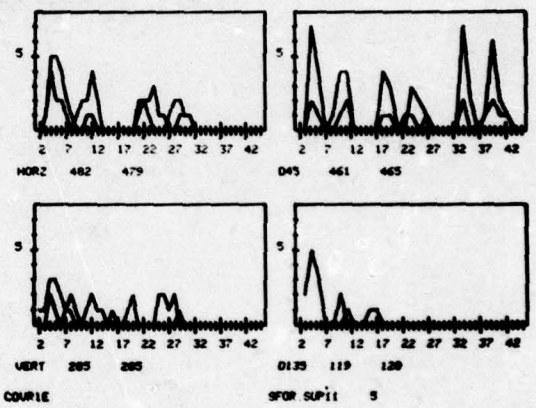


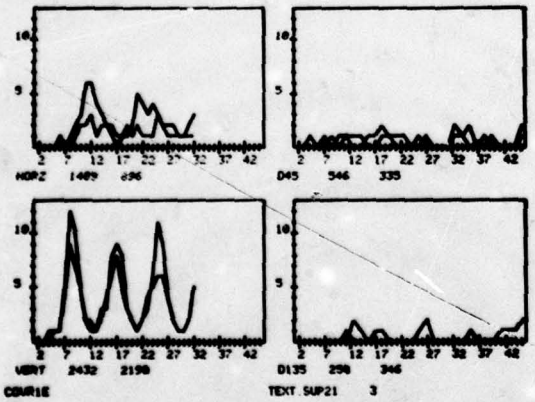
Fig. 6. Co-occurrence Matric Computation.



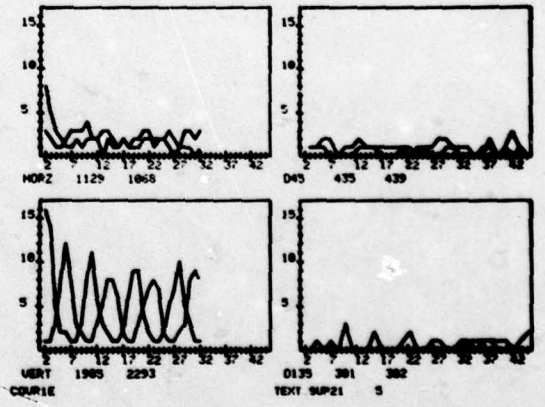
a) San Francisco Exact Edge Detection Matches Only.



b) San Francisco Opposite Edge Direction Matches Only.

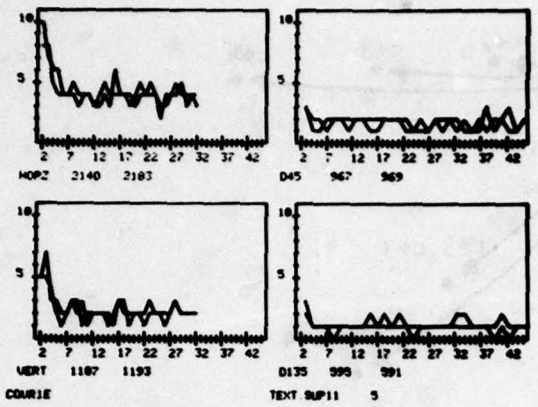


c) Raffia Exact Edge Direction Matches Only.

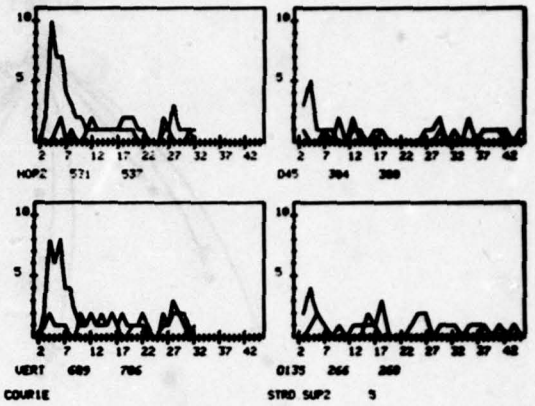


d) Raffia Opposite Edge Direction Matches Only.

Fig. 7. Co-occurrence Results Highly Regular Patterns.



a) Grass Opposite Edge Direction Matches Only.



b) Suburban Opposite Edge Direction Matches Only.

Fig. 8. Co-occurrence Results, Irregular Patterns.

BLOB EXTRACTION BY RELAXATION

Azriel Rosenfeld
Alan Danker
Charles R. Dyer

Computer Science Center
University of Maryland
College Park, MD 20742

ABSTRACT

A blob is a compact region lighter (or darker) than its background, surrounded by a smoothly curved edge. Relaxation can be used to enhance the blob's interior, i.e., initial "light" and "dark" "probabilities" can be made less ambiguous. Similarly, it can be used to enhance the blob's edge, i.e., initial (oriented) "edge" and "no edge" probabilities can be disambiguated. This paper uses two cooperating relaxation processes to improve both sets of probabilities by allowing them to interact with each other as well as with themselves; for example, "light" and "edge" mutually reinforce if the light point is on the light side of the edge. Results obtained in this way are better than the results obtained using either relaxation process alone.

INTRODUCTION

Segmentation of objects from their background is basic to many image understanding tasks. This paper deals with the relatively simple case in which the object is a "blob", i.e., a compact region, generally lighter (or darker) than its background, surrounded by a smooth edge. It points out how even this case cannot always be handled by elementary segmentation methods, and describes a compound relaxation process that makes use of both light/dark probabilities and edge/no edge probabilities to discriminate blobs from their background.

In principle, blobs can be extracted by thresholding the image at an appropriate level. However, if the image is noisy, thresholding will produce noisy results which may or may not be repairable by postprocessing. Moreover, thresholding may extract regions that are not bounded by edges, but are smooth continuations of the background, if the gray level fluctuations in the background happen to cross the threshold level.

Edge detection is sometimes useful in object extraction. Here again, however, there may be many edge detector responses in the interior of the object or background due to noise, and there may fail to be sufficiently strong responses on the object/background border due to blur.

Relaxation [1,2] has been used to improve the results of both thresholding [3] and edge detection [4,5]. To apply relaxation to thresholding, we initially assign "light" and "dark" probabilities to the image points based on their gray levels. We then iteratively adjust these probabilities at each point based on the probabilities at the neighboring points, i.e., light reinforces light and dark dark. This has the effect of shifting the probabilities initially assigned to noise points so as to make them more consistent with their surroundings. Eventually, the light probabilities at all points of a light region should become uniformly high, and vice versa, so that thresholding becomes easy, and should produce non-noisy results. Note, however, that the process may still extract regions that are not bounded by edges.

To apply relaxation to edge detection, we initially assign "edge" and "no edge" probabilities to each image point (or, alternatively, to each adjacent pair of points) based on the relative values of the gray level differences in various directions around the point. We then iteratively adjust these probabilities based on the probabilities at neighboring points: no edge reinforces no edge; edge reinforces edge if they smoothly continue one another, and reinforces no edge (and vice versa) if they are alongside one another. This has the effect of strengthening the appropriate edge probabilities at points that lie along smooth edges, and strengthening the no edge probability elsewhere, so that edge detection should yield less noisy results.

As we shall see in this paper, further improvement in the quality of the results is obtained if we use both the light/dark and edge/no edge relaxation processes, and also allow them to interact with each other. For example, "light" and "edge" at a pair of neighboring points reinforce one another if the light point is on the light side of the edge point, but they weaken one another if it is on the dark side. The details of this joint probability adjustment process will be given in the next section.

The joint (light/dark)/(edge/no edge) relaxation process can be regarded as making use of convergent evidence in (probabilistic) segmentation. Gray level and edge value are used jointly for segmentation in the (nonprobabilistic) "Superslice" scheme [6-8], among others. In this scheme

a set of thresholds is applied to the image; for each of these, the connected components of above-threshold points are extracted; and we call a component an "object" if there are many local maxima of edge strength around its border. Thus Superslice first makes two independent decisions based on gray level (thresholding) and edge strength (selection of maxima), and then checks them against each other; when they agree, we say that an object has been detected. The joint relaxation process described in this paper, on the other hand, never makes decisions; it estimates probabilities based on the gray levels and edge strengths, and then iteratively adjusts these probabilities so that both types of information are able to interact. This process thus combines the convergent evidence principle with the principle of deferred commitment. As Marr has pointed out, both of these principles are very desirable characteristics of an image understanding system.

LIGHT/DARK RELAXATION

Let g be the gray level of point P, and let b,w be the lowest and highest gray levels in the shape so that $b \le g \le w$ for all P. We take $p_w \equiv \frac{g-b}{w-b}$ as the estimate of the probability that P is white, and $p_b \equiv \frac{w-g}{w-b}$ as the probability that P is black.

The compatibilities between the black and white probabilities at adjacent points can be estimated by $\overline{p_ip_j}/\bar{p}_i\bar{p}_j$, where $i,j = b$ or w. Here $\bar{p}_i$ and $\bar{p}_j$ are the average probabilities of colors i and j, while $\overline{p_ip_j}$ is the average of the product of these probabilities taken over all pairs of adjacent points. These compatibilities can then be used as coefficients in the relaxation process described in [9]. (Alternatively, their logs, suitably rescaled, can be used as coefficients in the process described in [1]; the results obtained are essentially the same [3].)

Figure 4 of [10] showed the results of eight iterations of the relaxation process just described, applied to a FLIR image of a tank. In these figures, the probabilities have been redisplayed as gray levels, i.e., $g = b + p_w(w-b) = w - p_b(w-b)$. The histogram gradually turns into a pair of spikes at opposite ends of the grayscale, and the resulting discrimination between tank and background (it is not a segmentation, since we have not actually thresholded) is quite good.

It should be pointed out that in this simple scheme, if most of the points in the tank and the background had gray levels in the same half of the grayscale, the relaxation process would have driven them both to the same end of the grayscale. More general gray level relaxation processes which do not require that there be only two classes ("light" and "dark"), and do not depend for their success on restricting the ranges in which these classes lie, are described in [3].

Figure 1 shows another FLIR image of a blob and the results of applying eight iterations of the light/dark relaxation process to it. We see that the contrast between the blob and its background has been greatly enhanced, but that some of the light patches and spots in the background, which do not seem to be bloblike (i.e., they lack good edges) have also been somewhat enhanced. We shall next see what happens when the edge relaxation process, and then the joint process, are applied to Figure 1.

EDGE RELAXATION

Let $e_i (i=1,\ldots,8)$ be a measure of the gray level difference at point P in direction $45i°$. (In the experiments described below, we used the set of masks

$$\begin{matrix} -1 & 0 & 1 \\ -1 & P & 1 \\ -1 & 0 & 1 \end{matrix} \quad \begin{matrix} 0 & 1 & 1 \\ -1 & P & 1 \\ -1 & -1 & 0 \end{matrix} \quad \begin{matrix} 1 & 1 & 1 \\ 0 & P & 0 \\ -1 & -1 & -1 \end{matrix} \quad ,\cdots$$

to compute the e's.) Let E the largest e value at any point of the image, and let e_p be the largest value at the point P. We take $p_e \equiv e_p/E$ as an estimate of the edge probability at P, and $p_n \equiv 1-p_e$ as an estimate of the no edge probability. Moreover, we take $p_i \equiv (e_i/s)p_e$, where $s = \sum_{i=1}^{8} p_i$, as an estimate of the probability of an edge in direction $45i°$ at P; thus $\sum_{i=1}^{8} p_i = p_e$.

The pairwise compatibilities among $p_1,\ldots,p_8,p_n$ are estimated using ratios of average probabilities, as in the preceding section, and these are used as coefficients in the relaxation process of [9].

Figure 2 shows the initial p_e values for the same image as in Figure 1, rescaled as gray levels (i.e., $g = b+p_e(w-b)$), and eight iterations of the edge relaxation process applied to these initial values. We see that the edges of the blob are significantly enhanced, but some edges also begin to come out of the background.

JOINT (LIGHT/DARK)/(EDGE/NO EDGE) RELAXATION

Suppose now that we define compatibilities between p_b,p_w and $p_1,\ldots,p_8,p_n$ in the same manner as in the previous two sections. Given the two initial sets of probabilities for each point, we can then compute adjusted probabilties; but we normalize each of the two sets separately, i.e., at each iteration we divide the new estimates p_b' and p_w' by $p_b'+p_w'$, and the new estimates $p_1',\ldots,p_8',p_n'$ by $p_1'+\ldots+p_8'+p_n'$.

The results of doing this for the image used in Figures 1-2 are shown in Figure 3. This con-

sists of pairs of images, in one of which p_w is displayed as a gray level, and in the other p_e. We see that the use of the joint process has inhibited the growth of edges in the background (compare Figures 2 and 3). However, it has not entirely inhibited the emergence of white patches, not bounded by edges, from the background. This is because (understandably!) the no-edge probability has no inhibitory effect on either the light or dark probability.

Improved results are obtained if we initialize p_b and p_w differently, so that p_w is initially high only adjacent to edges on their light sides. (The idea that the human visual system "colors in" regions based on the gray levels adjacent to their edges is well known to perception psychologists.) Specifically, for each of the masks shown in the preceding section, we add the difference value e_i to the points marked by 1's. This is done for every mask in every position. The result is an array of "borderness" values which are high on the light sides of edges and low elsewhere.

We can now initialize p_b and p_w based on a combination of the gray level and the borderness value at each point. Let B be the maximum borderness value in the image, let β be the value at point P, and let $p_\beta = \beta/B$. Let $p_w^* \equiv \alpha p_w + (1-\alpha)p_\beta$, where $0 \leq \alpha \leq 1$, and let $p_b^* \equiv 1-p_w^*$. The results obtained when we use the initial values p_w^* and p_b^*, rather than p_w and p_b, for $\alpha = .25$ and $.5$, are shown in Figures 4-5, respectively. (Results for the light/dark process alone using these initial values are shown in Figures 4'-5'.) The coefficients are the same as those used earlier; only the initial values are different. We see that there is good improvement over the results in Figure 3, as regards inhibiting the emergence of light patches from the background.

POSSIBLE EXTENSIONS

This paper has illustrated the advantages of allowing interactions between relaxation processes. This approach can be regarded as using convergent evidence at the probabilistic or fuzzy level, prior to making any commitment to a firm decision.

An alternative idea, currently under investigation, is to use "inside" and "outside" labels in addition to the "light" and "dark" labels. Initially, most points would have equal probability of being inside or outside (assuming we do not know whether the blobs are light and the background dark or vice versa), except that points adjacent to an edge on the side away from the center of curvature have higher probability of being outside, while those on the side toward the center of curvature have higher probability of being inside. These probabilities can then reinforce one another, i.e., inside reinforces inside and outside reinforces outside for a neighboring pair of points, to the extent that they are not separated by an edge. Initially, the probabilities at a given point of the blob border will depend on whether the border is convex or concave at that point; but eventually the interior of the blob should be uniformly labeled "inside", and the exterior "outside", with high probability. Note that this method can also be used to label the interior of a closed curve, even if it does not differ in gray level from the exterior. Processes of this type might be used to model figure-ground ambiguity (the Rubin vase). Similarly, processes involving light/dark labels might be used to model pseudoedges (the Craik-O'Brien-Cornsweet phenomenon) and virtual edges.

An interesting possible extension would be to use a (light/dark)/(edge/no edge) relaxation process at several different resolutions (e.g., at each level of a "pyramid"), and allow the levels to interact. Note that blobs look like (local) spots at the appropriate level, so that immediate interaction between the blob interior and the edges on all sides of it becomes possible; in the process described in this paper, on the other hand, a considerable number of iterations is required in order for the parts of a large blob to reinforce one another. Thus extending blob-detection relaxation to a pyramid representation would permit fast interactions among all parts of the blob; this is an example of the speed advantage of a pyramid cellular processor over an ordinary cellular array. Similar remarks apply to the inside/outside labelling process.

The pyramid extension, which is currently being explored, also provides some other possible benefits. A compact blob maps into a spot at some level of the pyramid, while an elongated (but relatively straight) blob maps into a line segment. By suitably designing the interlevel interactions, one could bias the process to prefer elongated blobs over compact ones, or vice versa. This would provide an interesting basis for using primitive types of shape information from the very beginning of the segmentation process, rather than performing general-purpose segmentation and then rejecting regions that have the wrong type of shape. Conceivably, multilevel interactions in a pyramid may also provide a basis for (fuzzy) hierarchical representation of complex shapes, but it is not yet clear how to do this. Some other mechanism than a pyramid would certainly be necessary for representing the shapes of curves.

Interactions among relaxation processes may also be useful in processing multiple images, e.g., in detecting objects in steopairs or in time sequences of images. This would involve iterative estimation of disparity or velocity at each point concurrently with the estimation of lightness/darkness and edgeness. Some work

on the time sequence applications is currently in progress, and will be reported elsewhere.

REFERENCES

1. A. Rosenfeld, R. A. Hummel, and S. W. Zucker, Scene labeling by relaxation operations, IEEE Trans. SMC-6, 1976, 420-433.

2. A. Rosenfeld, Iterative methods in image analysis, Pattern Recognition 10, 1978, 181-187.

3. R. Smith and A. Rosenfeld, Thresholding by relaxation, Computer Science Center, University of Maryland, Technical Report, in preparation.

4. B. R. Schachter, A. Lev, S. W. Zucker, and A. Rosenfeld, An application of relaxation to edge reinforcement, IEEE Trans. SMC-7, 1977, 813-816.

5. A. R. Hanson and E. M. Riseman, Segmentation of natural scenes, in A. R. Hanson and E. M. Riseman, eds., Computer Vision Systems, Academic Press, New York, 1978, 129-163.

6. D. L. Milgram, Region extraction using convergent evidence, Proc. Image Understanding Workshop, April 1977, 58-64.

7. D. L. Milgram, Progress report on segmentation using convergent evidence, ibid., October 1977, 104-108.

8. D. L. Milgram, Region extraction using convergent evidence, TR-674, Computer Science Center, University of Maryland, June 1978.

9. S. Peleg, A new probabilistic relaxation scheme, TR-711, Computer Science Center, University of Maryland, November 1978.

10. Project Status Report, in these proceedings.

ACKNOWLEDGEMENT

The authors wish to thank Russ Smith for implementing the "borderness" scheme.

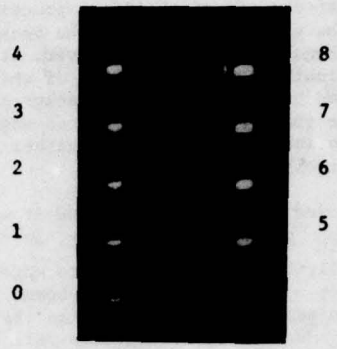


Figure 1. Eight iterations of the light/dark relaxation process.

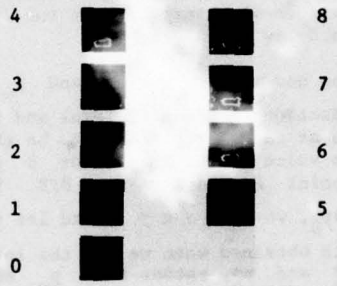


Figure 2. Eight iterations of the edge/no edge relaxation process.

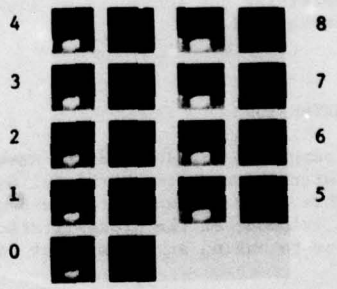


Figure 3. Eight iterations of the combined process.

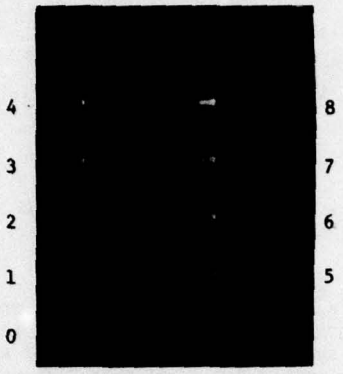


Figure 4. Same as Figure 3, but using initial values based .25 on gray level and .75 on "borderness" (see text).

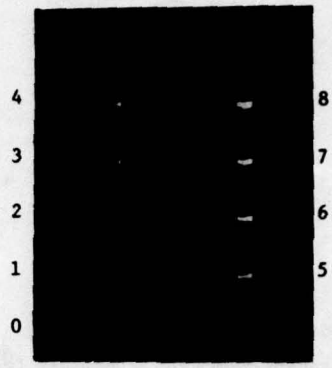


Figure 4'. Same as Figure 4, for the light/dark process only.

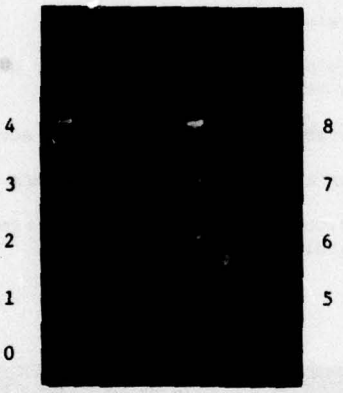


Figure 5. Same, but using (.5,.5) initial values.

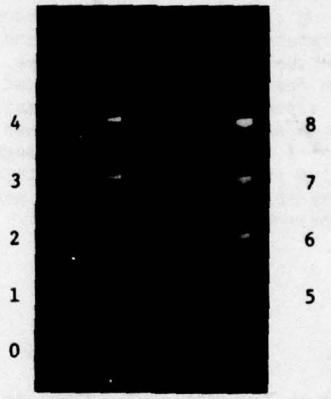


Figure 5'. Same as Figure 5, for the light/dark process.

KNOWLEDGE-BASED DETECTION AND CLASSIFICATION OF VEHICLES AND OTHER OBJECTS IN AERIAL ROAD IMAGES

Gerald J. Agin

SRI International
Menlo Park, California 94025

ABSTRACT

We describe a set of techniques for identifying and classifying vehicles in aerial images of road scenes. The approach is knowledge-based and draws on three sources of knowledge: generic knowledge about the domain, a data base containing information specific to the site, and data associated with the image itself. Understanding shadows is crucial to successful scene understanding in this domain, and we present three techniques for dealing with them. After a correlation road tracker locates a road and identifies visual anomalies, the anomalies are examined by a set of expert subroutines that seek to establish if the anomaly is the image of a vehicle (plus its shadow) or of something else. Vehicles may be separated into types according to their dimensions.

I INTRODUCTION

One of the overall goals of the SRI Image Understanding project is to explore the ways knowledge can be used on problems of interpreting aerial imagery [1]. It has been demonstrated repeatedly that the more knowledge is available and used in artificial intelligence programs, the better the results tend to be. The ideal system should be flexible enough to use the information that it has, but also to be able to function, albeit with somewhat reduced capabilities, if some of the information is not available.

This paper describes an approach to finding and identifying vehicles in aerial images, using diverse sources of knowledge. The following traffic-monitoring scenario provides a domain for this work: Given a digital aerial image and a data base, the problem is to detect vehicles on the road and to classify them as to vehicle type. The image should have sufficient spatial resolution to allow recognition (about one foot per pixel, minimum). Figure 1 shows a typical image of an area containing a freeway interchange.

The data base contains information about some limited geographical area of interest. As a minimum, it should have the locations of known roads in the area. Other relevant information could include (but not be limited to):

- Road width
- Brightness profiles across the road
- Terrain information
- Buildings, railroads, and other cultural features
- Intersections, overpasses, and access roads
- Signs and permanent road markings
- Previous photo coverage of the area, in digital form.

Figure 1 An Aerial Road Image

A calibration procedure [2] establishes correspondence between image coordinates and geographic coordinates, allowing us to convert quickly back and forth between coordinates in the data base and pixel locations in the image. A road tracker [3] uses the road location predicted by the data base to trace the road centerline and boundaries by correlating successive profiles perpendicular to the road direction. Areas where the image disagrees with the expected road profile are identified as "anomalies." These areas are passed to the classification routines for further scrutiny.

Many different conditions could give rise to an anomaly. Vehicles usually show up this way, but so do the shadows of objects off the road (trees, buildings, signs, utility poles), overhanging trees, painted markings on the road, and changes or irregularities in the road surface (such as tar patches). There are also some less usual situations with which a practical system ought to deal, such as road construction, floods, bomb craters, smoke, and dust clouds. The classifier must first decide if the anomaly arises from a vehicle or from some other cause. Only then can it proceed to classify the vehicle type.

Although the scenario assumes some rather specific resources and goals, the knowledge-based approach we have developed for vehicle classification is generally applicable to a wide range of object-recognition tasks in cartography and photo-interpretation.

II SOURCES OF INFORMATION

A wide variety of information can be helpful for detecting and classifying vehicles. We can identify three kinds of knowledge relevant to this problem: knowledge about the problem domain (generic knowledge), knowledge about the site (the data base), and knowledge about a particular place and time (information associated with the image).

Generic knowledge includes information that can be deduced from functional descriptions. A road is a narrow, linear region upon which vehicles may travel. The road is usually continuous in the image--if it appears discontinuous it may be that there are obstructions, or there may be shadows or discolorations on the road surface. Roads have minimal variation in the direction of travel but may have considerable variation in the perpendicular direction, because of the different compositions of road bed, shoulders, and an expected pattern of oil stains in the center of each lane. We have some idea of the expected shapes of vehicles viewed from different angles, and an expectation that they probably will be aligned parallel to the road direction. Our illumination models take into account the physics and geometry of shadows, and we can sometimes use shadows to make inferences about objects. We know the usual places where road signs, utility poles, and painted road markings are located. All the foregoing can be used to make sense out of a road scene.

The data base is a useful source of information. Its principal use is to predict the approximate road centerline so that the road tracking subroutines can operate. But other kinds of information can be brought into play. Terrain information can be used to refine position estimates when the viewing angle is not vertical, and to predict shadows better if the ground slopes. Classifying shadows of objects off the road is very much simplified when it is known what objects are likely to cast shadows. Ambiguous anomalies in the image can sometimes be distinguished if a picture can be compared with a previous one or, better yet, if the data base states what anomalies were found in previous images and how they were classified. Intelligence reports and expected traffic conditions can help the program decide what to look for or what strategies to use.

The greatest single source of data is the image itself. It is easy to overlook some information that is associated with the image but may not be in the actual raster. For example, it is usually possible to ascertain (at least approximately) the altitude, position, and heading of the aircraft from which the image was taken. Scaling parameters, view angles, and compass headings can be derived by calibration. If the time and date the picture was taken are known, the sun position can be calculated, but even without these data the sun position usually can be estimated from shadows.

In short, detection and classification of vehicles is not based solely on what is in the image. In the following sections, we detail some of the ways we use the available information.

III USE OF THE CORRELATION ROAD TRACKER

We depend on the correlation road tracker designed by Quam [3] to isolate anomalies in images or roads. These are regions where attention should be focused.

The road tracker is based on the assumption that variations in road surface materials, centerlines, and intralane wear patterns correspond linearly to the road itself. Vehicles and other anomalies, however, stand out as being quite different from the pattern of the road. Detecting these anomalies is important to the operation of the road tracker. Where substantial disagreement occurs between successive profiles, the corresponding pixels are marked as anomalies, so that these points can be eliminated from the correlation calculations. If the anomalies were not so masked, they would perturb the location of the correlation peak and introduce errors.

Figure 2a shows a representative excerpt from the area covered by the image of Figure 1. The road tracker is initiated by specifying a single profile approximately perpendicular to the road direction and centered on it. This initial baseline is selected manually now, but facilities exist for using the data base to draw the baseline automatically.

The road tracker produces several forms of output. As indicated by Quam [3], the program can produce a point list describing the track of the road center, and a binary image of all points in the road that are anomalous. But for vehicle identification, another form of output has been added. The road reflectance model may be subtracted from each pixel considered, resulting in a difference image that has the road profile subtracted out. Figure 2b shows the baseline, the road center, and anomalies detected. Figure 2c shows the difference image.

The grey-scale difference image may be converted to a binary anomaly image by thresholding. Although we now use a threshold identical to the one used by the road tracker to produce the anomaly image, the threshold value could be adjusted as a function of various considerations, such as success or failure of a previous analysis.

In the difference image, shadows tend to have a relatively uniform intensity, even though the road reflectance profile varies considerably. If we adopt the simplifying assumptions that any object casting a shadow may be approximated by a half plane of infinite extent that hides all but a fixed proportion of the sky, and neglect reflected illumination from nearby objects, then the ratio of intensities across the shadow edge should not depend on the reflectivity of the underlying surface. When the original image is digitized on a logarithmic brightness scale, this constant ratio becomes a constant intensity in the difference image. Because the assumptions are approximate at best, the constant-difference test is almost never exact. Nonetheless, by subtracting the road profile from the image, we can expect the intensity of shadows to be more uniform in the difference image than in the original one.

On the other hand, when anomalies are caused by vehicles, subtracting the road profile will cause its inverse to be superimposed on the anomaly. Figures 3a and b show an original image and a difference image (from another road site) that demonstrate these peculiarities. Both kinds of image are useful in classifying anomalies.

As the road tracker proceeds, it constantly keeps track of the average correlation between successive road profiles at their optimum locations. This correlation value, a useful estimate of noise in the picture, is made available to succeeding classification stages.

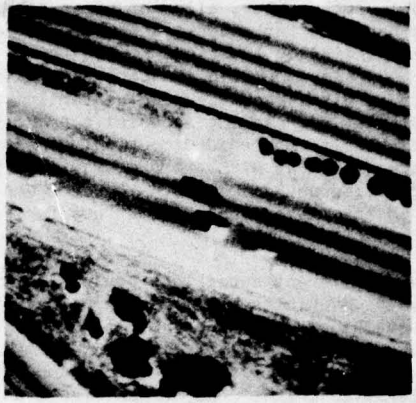


Figure 2a Road Scene

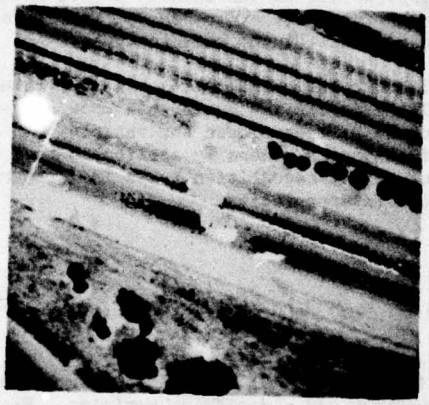


Figure 2b Baseline, Centerline, and Anomalies

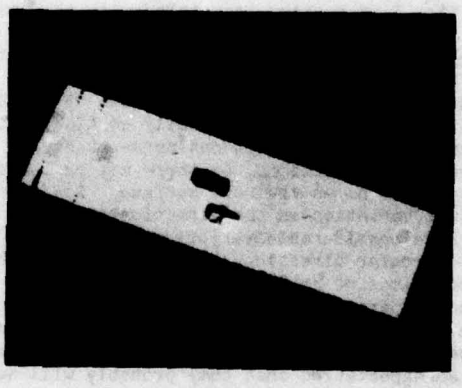


Figure 2c Difference Image

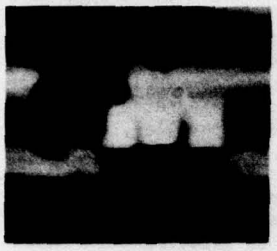


Figure 3a
Original Image

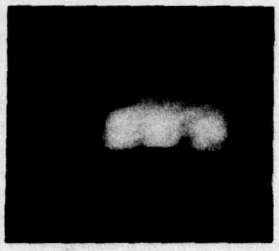


Figure 3b
Difference Image

IV SHADOWS

An understanding of shadows is crucial to making sense out of high-resolution aerial images. The scene is always out-of-doors, and is usually illuminated by direct sunlight because photoreconnaissance missions are flown mainly in clear weather during the day. Sunlight produces deep, dark shadows. Frequently shadows are the most prominent visual feature of an image.

For vehicle classification, many of the anomalies the classifier is called on to consider are only the shadows of objects off the road, such as trees, signs, or utility poles. All vehicles cast shadows, and, unless the boundary between the vehicle and its shadow can be determined, classification on the basis of shape is hopeless. Furthermore the existence or nonexistence of a shadow can aid in deciding whether or not a given anomaly is a vehicle. The size and shape of the shadow can give valuable clues of the height of the vehicle and its profile. As a dramatic demonstration of this, consider the vehicle shown in Figure 4. Because the reflectance of the vehicle is almost the same as that of the road, the vehicle might have gone un-noticed were it not for the shadow. But the shadow not only gives away its position, it tells us the vehicle is probably a Volkswagen "beetle."

We have a number of techniques at our disposal for identifying shadows. The simplest is based on the brightness model. The technique is simply to search for all pixels in the image whose intensity is in the range of values expected for shadows. This works somewhat better in the difference image than in the original, because the effects of variation in the road surface are reduced. Figure 5 shows the central portion of the area analyzed in Figure 2, which we will use to illustrate shadow-finding techniques. Figure 6 shows the shadows extracted from Figure 5b by this method.

In our work so far, the expected range of shadow intensities has been inferred from the statistics of areas manually indicated as shadows. It should be possible in principle to automate this procedure, for example by using the using the data base to predict or find known shadows. Alternatively, it seems likely that a formula can be derived that will give the expected distribution based on calibration of photometry.

In situations in which the correlation road tracker is not applicable, shadows located by the brightness model might indicate areas of the picture that deserve scrutiny.

Figure 4 Vehicle with Shadow

Figure 5a
Original Image

Figure 5b
Difference Image

Figure 6 Shadows Found by Brightness Criterion

Another device, based upon a predictive model, depends on knowing the sun angle. The shadow of any raised object is always on the side away from the sun, and if the height of the object is known, the width of the shadow can be predicted. Figure 7b shows the areas identified as shadow from the image of Figure 5b by thresholding the difference image to locate anomalies, and assuming each anomaly to be due solely to an object five feet tall plus its shadow.

Figure 7 Shadows Found by Predictive Criterion

The third technique is based on a projective model. It tries to look directly for the shadow edge. Vehicles tend to be rectangular when viewed from above, and, unless the sun is directly ahead of or behind the vehicle, there will be a long, straight edge separating the vehicle from its shadow. This edge can usually be found by performing a Hough transform [4] on the gradient of the image, or (equivalently) by projecting the gradient onto axes oriented in various directions and finding the direction from which the gradient points tend most to reinforce each other. However, much better results are obtainable when the direction of the edge is known or assumed a priori. Such is usually the case, for vehicles tend to be oriented parallel to the road direction.

An example of shadow detection by projection is presented in the next section.

The three techniques are based on different sets of assumptions and are applicable in different circumstances. The projective method is useful only for finding shadows of vehicles. The predictive model is more generally useful, being applicable to objects off the road as well as on it. The brightness model makes no assumptions about the object casting the shadow--it only requires that the background on which the shadow is cast be relatively uniform.

V CLASSIFICATION OF ANOMALIES

For classifying anomalies, we have chosen to construct a number of "expert" subroutines, each of which tests a specific hypothesis. For example, the vehicle expert determines whether or not a given anomaly could be a vehicle (plus its shadow), and if so, attempts to say whether the vehicle is a car or a truck. The tree-shadow expert tries to say whether or not the anomaly could be the shadow of an object off the road, and the road-marking expert similarly looks for painted markings. Other expert modules could easily be integrated into the scheme. The experts operate in parallel, each expert forming its decision without interacting with the other experts. The top level program chooses the most likely interpretation of the anomaly. If no expert subroutine is able to account for the anomaly, it is labelled "unclassified."

The vehicle expert is the most involved of the expert subroutines. It first examines the overall size (area) of an anomaly. If the anomaly is too small or too large, it is rejected. Next, a search is made for long edges that might be the sides of the car, by projecting the gradient image to a baseline. For the projection a binary mask is used so that only those points near the anomaly are considered--the mask is generated by expanding ("growing") the anomaly three pixels. Figure 8a shows the results of applying a gradient operator to the image of Figure 5a. The masked gradient was projected on the axis drawn in Figure 8b, where the average projected gradient magnitude is plotted.

Figure 8 Use of Projection to Find Shadow Edges

A line perpendicular to the direction of the road is used as an initial baseline. If some evidence of edges is found, small perturbations are made to the orientation to find a local maximum. If the edges are not found, a global search is made for a direction of projection that will show the edges. If the edges are not found again, the anomaly is rejected.

Note that there are three peaks in the plot, corresponding to the boundaries between road and car, between car and shadow, and between shadow and road. The three highest peaks in the projected gradient are examined to see if they are in the correct relationship. Average brightness is projected to the same baseline to see if the brightness of the shadow portion is appropriate. A figure of merit is computed from these tests, indicating the closeness of measured spacing and brightness to the expected spacing and brightness. The figure of merit is used later in choosing the most likely interpretation of the anomaly.

The average width of the shadow and the location of the sun may be used to estimate the height of the vehicle. A tolerance or range of uncertainty is also computed at this time, because the combination of low spatial resolution and a disadvantageous sun angle may make the height figure not particularly useful. A nominal height of 6 feet is used for predicting a shadow to the front or the rear of the vehicle, and this predicted shadow length subtracted from the length of the original anomaly gives the length of the vehicle.

Classification as to vehicle type is relatively crude at this time. If the overall length of the vehicle is greater than 20 feet, or if the height can reliably be stated to be greater than 6 feet, the vehicle is called a "truck". Otherwise it is called a "car."

Another expert subroutine identifies shadows of objects off the road. To qualify as such a shadow, an anomaly must have an average brightness lower than the average road brightness, and extend to the edge of the road on the side nearer the sun. A figure of merit is calculated from how well the average brightness (in the difference image) corresponds to the predicted value, and from the variance of brightness inside the anomaly.

The expert on painted markings on the road is similar to the shadow expert. Painted markings are always brighter than the road surface and limited in total area. The figure of merit is based only on variance of brightness; a much lower variance is expected for road markings than for shadows.

VI DISCUSSION

The state of our experiments in anomaly classification is such that it is too early to report any quantitative results. However we can say, qualitatively at least, that the methods outlined above succeed on the easy cases and break down on the difficult ones. We have tested our programs on approximately 20 different scenes extracted from three diverse road areas. Where good contrast exists between an anomaly and the road, and (in the case of vehicles) the shadow is visually distinct from the object casting it, we have little difficulty in obtaining a correct identification. Where conditions are not as good, the programs tend to fail to make any identification rather than to come up with a misclassification. Additional robustness in the classifier will be necessary to enable it to handle unusual cases.

The various expert subroutines are not now integrated in any way. Each reports its figure of merit to the top-level program, which selects among the hypotheses. A more useful system should allow interaction among the various experts.

Figure 3 shows a good example of a case that could be handled by cooperation of the tree-shadow and the vehicle experts. It might be sufficient if the shadow expert were to realize that it could interpret part of the anomaly, subtract the explainable part, and ask the other experts to classify what remains. The vehicle expert would have to take the situation into account and not look for a separate shadow for this anomaly.

Figure 9 is difficult to analyze without higher-level knowledge. A more direct link to the data base would be particularly useful in this case, enabling us to separate the anomaly into portions that are "expected" (the visible portions of the arrow) and "not expected" (the car and its shadow).

Much generic knowledge tends to be expressed in the coding of the computer programs that analyze pictures. In this form it is inflexible--adding new knowledge involves writing new computer programs. A long-range goal of this research is to find new ways of expressing this kind of information, for example in the form of rules or templates. Such a capability would lead to highly competent computer visual capabilities that would greatly enhance interactive and automatic cartograhpy and photo-interpretation.

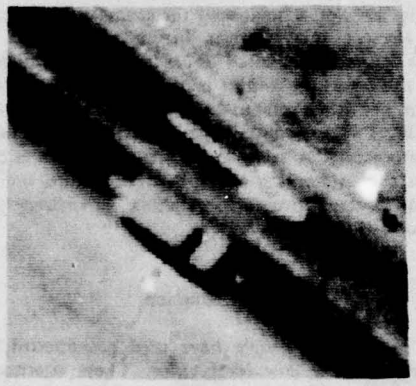


Figure 9 A Vehicle over a Road Marking

REFERENCES

1. M. A. Fischler, et al., "The SRI Image Understanding Program," Proceedings: ARPA Image Understanding Workshop, Menlo Park, Calif., (April 1979).

2. R. C. Bolles, et al., "The SRI Road Expert: Image to Database Correspondence," Proceedings: ARPA Image Understanding Workshop, Pittsburgh, Pa., (November 1978).

3. L. H. Quam, "Road Tracking and Anomaly Detection in Aerial Imagery," Proceedings: APRA Image Understanding Workshop, Cambridge, Mass., (May 1978).

4. R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," Communications of the ACM, V. 15, No. 1 (January 1972).

GOAL-DIRECTED EDGE LINKING AND RIBBON FINDING

Rodney A. Brooks

Artificial Intelligence Laboratory, Computer Science Department
Stanford University, Stanford, California 94305

Abstract

We describe an algorithm for linking straight edge elements produced from a bottom up line finding stage. Edges are linked by a best first search algorithm. Heuristics are used both to select candidate edges for linking and to prune the search tree. A list of chains of edges is the resuk of this stage of processing. The choice of heuristics determines the chains of edges produced. Further heuristics prune the list of chains. Finally regions, described as ribbons, are chosen so that their boundaries are approximated by the chains of edges. Individual picture elements may appear in multiple ribbons. Further heuristics can be employed to reduce this multiplicity. We describe the algorithm and its implementation. It is organized so that a higher level computational procedure can easily direct this low level algorithm by supplying and altering the heuristics during processing. A selection of useful heuristics is described and then shown working on an example picture, with a hand simulated control program, resulting in extremely useful descriptions of the image.

Introduction

A number of people have used goal-directed heuristic search methods for low level vision. These algorithms have primarily dealt with pixel data. The heuristics used have been either fixed, or dependent on a few parameters. Some of these parameters are altered depending on image quality, and others depending on the contour being sought.

Many of those using these approaches have tried to locate ribs and tumors in radiographs of human chest cavities. For instance, Ashkar and Modestino [1978] use a standard best first tree search algorithm (they call this the "Zigangirov-Jelinik stack algorithm") to follow contours. The branches of the tree arise from examining the pixels neighboring the node pixel which are not already members of the path. The metric, or evaluation function used, measures the continuity of the contour, smoothness of curvature and a comparison to a prototype contour. The weighting of these three factors is altered depending on the noise in the image - relying less on the prototype for cleaner pictures. Since their example domain is finding human ribs in x-ray images, they can hopefully be confident of an image uncluttered with unexpected objects. This method relies on the existence of either good quality uninterrupted contours, or good predictions of the contour.

Ballard and Sklansky [1976] look for tumors, and Wechsler and Sklansky [1977] for ribs. In both cases they obtain an approximate area for the object of search, then use a modified depth-first search method to follow contours. The best successor pixel is taken at each stage of the search. Both these systems rely on powerful techniques heavily dependent on the application to constrain the search.

Martelli [1972] formulates the problem of edge following as a shortest path graph search and uses the traditional A* algorithm to carry it out. In Martelli [1976] he shows how *a priori* knowledge of the shape of the contour can be embedded in the cost function to constrain the search.

In this paper we will describe a system which uses heuristic search to link edge elements (rather than pixels) into contours, and further heuristics to evaluate the worth of contours so produced. The system is quite general, and relies on the heuristics supplied to constrain the problem. It is intended that the system be controlled by other sections of a general purpose vision system (the ACRONYM model-based system, see Brooks, Greiner and Binford [1978a], [1978b]). Thus it needs to be both powerful and easy to program.

We deal with unregistered images of cluttered scenes. Simple bottom up procedures are not adequate to deal with such a situation. In a typical example the algorithms we describe here will be invoked many times, by a higher level program, which will change the heuristics and associated parameters, depending on its current belief of the scene contents, based on the previous results from the edge linking routines.

The edges we use as data come from segment files produced by Nevatia and Babu [1978]. They use directional masks to detect edges which they thin, threshold and fit with piecewise linear segments. The linear pieces are directed such that their right sides are brighter than their left in the original image. Further they are often linked into longer consistent edges, which are called supersegments. We will refer to individual linear pieces as edge elements or segments. Figure 2 shows edge elements produced by Nevatia and Babu from the photo in Figure 1.

The Edge Mapping Algorithms

In this section we will outline the algorithms we have developed for extracting region descriptions from edge data. These algorithms are goal-directed and need a higher level system to supply them with heuristics. We will refer to the higher level system as the executive. The algorithms we describe

here are all implemented and running in MACLISP. They form the prototype edge mapping module (we will occasionally refer to it as EM) of the ACRONYM model-based vision system (see Brooks, Greiner and Binford [1978a, 1978b] and Binford and Brooks [1979]). The EM will have its heuristics supplied by, and be directed by the predictor and planner module of ACRONYM, via the matcher. We defer the discussion of particular heuristics which might be used until the next section. Often the heuristics mentioned below are described as predicates. It should be noted that they can be parameterized via global variables.

There are four relatively independent phases of the algorithm; linking candidate contours, discarding unsatisfactory candidates, finding ribbon descriptions for contours and the regions they bound and finally disambiguating redundant descriptions of a single part of the two dimensional image. We discuss these four phases in order below. In general we expect that for a given goal, the first three of these phases would proceed in a serial manner, although the results from a single invocation of the phase of linking candidate contours may be used by multiple invocations of the next two phases, with varying goals. The last phase may occur as part of the actual matching process; we will however demonstrate a useful heuristic, in the next section, for use in a bottom up situation.

Fig. 1. Digitized photograph.

Linking Candidate Contours

Linking edges into a contour can easily be formulated as a tree searching problem. Each node is an ordered list of edge elements, with a score attached giving a measure of goodness of the contour specified by the list. There is also an optional direction. Recall that the edge elements are directed so that their right sides are brighter than their left. A direction of "right" will be interpreted to mean that the edge list is following around a relatively brighter region, and "left" a darker region. The root node of the tree is a list of a single edge element. A descendent node m of a node n with edge list $(e_{n1}, e_{n2}, \ldots, e_{nk})$, will have edge list $(e_{n1}, e_{n2}, \ldots, e_{nk}, e_{nk+1})$; i.e. each descendent node extends the contour by one new edge element. If a node has a direction then its descendent nodes have the same direction. In its most general form, each node has a descendant for every edge element in the image which is not already on its edge list. To find the best contour starting with a given edge element we must search the tree to find the the node with the best score.

There are four issues to resolve for such a tree search.
1. Which edges should be chosen as root nodes of search trees?
2. The tree as defined will clearly be enormous (of size $(n-1)!$ where n is typically 500 to 1000). Therefore the tree must be pruned, and not all nodes searched. How should this be done?
3. What scoring function should be used?
4. In what order should the tree be searched and when should the search be terminated?

The solutions to these four problems incorporate a capability for the executive to direct the contour finding process.

To choose root nodes for the search trees, the executive supplies a list of candidate egde elements and a list (called *root-tests*, perhaps empty) of predicates on edge elements. An edge which satisfies any of the *root-tests* has its search tree expanded. Note that there is some redundancy in the two mechanisms. This is to provide efficiency. The predicate list provides a uniform selection procedure, and there is a single instance of the control structure within the low level edge mapping module. In many cases the executive will simply provide a list of the edge elements to be pruned by the *root-tests*. However suppose the executive is only interested in a restricted area of the total image and the edge elements are hashed on their two dimensional location. It will then be more efficient for special purpose routines within the executive to provide a list of location specific edges, rather than use a predicate within *root-tests*, as the latter can not benefit from the having the edges hashed - it must test each edge in turn.

We use a best first search algorithm to search partially a sub-tree of the total search tree. The sub-tree which may actually be searched is defined by two lists of heuristics provided by the executive; the *producers* and the *reapers*. As we will see below the ordering of *producers* is important for the order in which the search tree is traversed, and so in the case of *incomplete search* (the normal case), this ordering can effect the search outcome. The ordering of *reapers* can effect the efficiency of the the search but not the outcome. The depth first search, combined with the scoring function determines the search order.

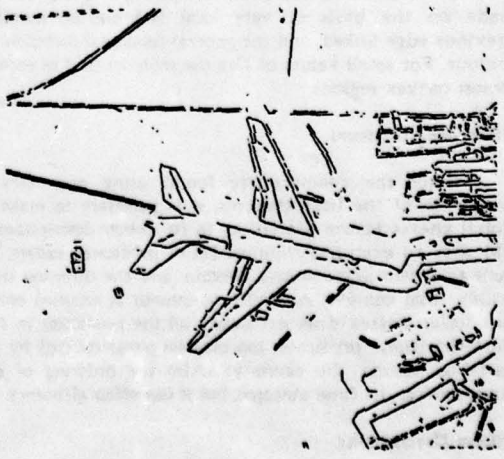


Fig. 2. Lines found in image.

A best first tree search algorithm works as follows. A list of nodes is kept in decreasing order of their score. Initially the list contains only the root node, which is assigned score zero. A node is said to be fully developed if all its descendants in the sub-tree defined by the *producers* and *reapers* have already been placed on the the ordered node list. A node is partially developed if it is not fully developed. Best first search finds the first partially developed node on the ordered list (i.e. the partially developed node with the highest score), calculates another descendant and its score, and inserts the new node into the list. This continues until there are *n_nodes* (a parameter supplied by the executive) fully developed at the head of the list. The result of the search is the edge list associated with the node at the head of the list, which will be the highest scoring node found during the search. Note that a node can never be promoted on the ordered list. Further, the search for a node to develop will never proceed past the first *n_nodes*, for then there would have to *n_nodes* fully developed, and so the tree search would terminate. Thus only the first *n_nodes* of the ordered list need be retained.

The *producers* are functions of an edge element and a direction, used to produce descendants of a node. They are applied to the most recent edge of a node, and the node's direction. A producer returns a list of descendants, and a score for each of those edges. When the search procedure wants to develop a node, by finding another descendant, it checks to see if there are any edges left from the last producer used. If not it calls the next producer. It continues until it has an edge element which is neither already in the contour, nor has been previously suggested by a producer. It also demands that if the parent node had an associated direction, the exterior angle made by the new edge element is within *slop* degrees of continuing in that direction (*slop* is a parameter supplied by the executive). Finally the search procedure tests the proposed new edge element with the predicates in the list *reapers*. These are predicates of edge list from the parent node, the proposed new edge and the direction of the parent node. When the search procedure finally has a new edge it makes up a new node, with that edge added to the contour, and a score from the parent node, incremented by the score associated with the new edge.

Thus the decisions about which edges to link are mostly made on the basis of very local information, namely the previous edge linked, and the general rotational direction of the contour. For small values of *slop* the contours tend to encompass almost convex regions.

Discarding Contours

Since the contours are found using only very local properties of the edge elements, it is necessary to make more global checks before proceeding to fit ribbon descriptions. The EM uses an executive supplied list of predicates, *cullers*, which each take two arguments; a contour and the direction used in finding that contour. A candidate contour is retained only if it and its associated direction satisfy all the predicates in *cullers*. Note that these predicates too may be parameterized by global variables set by the executive. Also the ordering of *cullers* cannot effect the final outcome, but it can effect efficiency.

Ribbon Descriptions

Ribbons are planar area descriptors which are the two dimensional analogue of Binford's [1971] generalized cones. We describe the area defined by a contour as a ribbon. We currently use a subset of the general definition of the class of all ribbons. In particular, they must be defined by sweeping a symmetric width element normally along a straight spine while changing the width linearly with distance swept. We will call the two lines defined by the width element as it is swept along the spine, ribbon edges. Thus the ribbons we use can be fully specified by a line segment (the spine) and a width at each end.

Fitting a ribbon to a contour is currently done by a fixed algorithm with no mechanism for goal direction from the executive. It proceeds as follows. A histogram of the angles of the edge elements making up the contour (weighted for edge length) is constructed at 20 degree intervals. The two peaks with the largest areas, between local minima, are found, and the edge elements which contribute to each are identified. The two edges of the proposed ribbon description are fitted to these collections of edge elements. First the mean angles (weighted for edge element length) are calculated, and then straight lines, constrained by these angles are fitted by least squares, weighted by length once again. The line whose angle is the mean of the ribbon edge angles, and which is equidistant from the ribbon edges, is calculated. The edge elements which defined the ribbon edges, are normally projected onto the center line, to define the extremities of the spine. The width function for the ribbon can then be easily calculated from the spine and the ribbon edges.

Disambiguating Ribbons

Often there will be multiple ribbon descriptions constructed for essentially the same region of an image. The EM provides no explicit mechanism to apply heuristics to this problem. The disambiguation problem requires comparisons amongst ribbons, and is thus essentially an n^2 process, for n ribbons. For individual heuristics, large savings in the constant of proportionality can be made, by saving partial results between the n^2 comparisons (e.g. in the example given later, it is prudent to precalculate all the centers of gravity). We have not found a convenient mechanism to handle the most general case efficiently, thus we have left the problem to special purpose programming by the executive.

Some Examples

In this section we will discuss some heuristics which can be used to direct the EM towards prescribed goals. It should be noted that the goal direction is not explicit, but rather implicit in the choice of heuristics and their parameters. We will use as examples here, the image of an L1011 airplane on the ground at San Francisco airport shown in fig. 1, working with the edge elements shown in fig. 2, and edge elements derived from an aerial photograph of an airfield, shown in fig. 3. We will consider the pictures to be 1 unit square in the following discussion.

To increase the efficiency of the heuristics we hash the edge elements on their angles at 1 degree intervals, and on a list of their angles at 20 degree intervals and the x and y coordinates of the edge beginning, to a resolution of one sixteenth of the image. Only edges which are at least five pixels long in the original image (512 by 512 pixels) are hashed. This restricts the number of edge elements which might be considered at all to 477 for the airplane scene and 170 for the airfield scene.

The only *root-test* we used in these experiments passed any edge element whose length was greater than some threshold, or which had a successor in some super segment. Using a threshold of 0.03 and considering all the edge elements, this results in 232 search trees for the airplane scene, and 196 for the airfield scene being expanded.

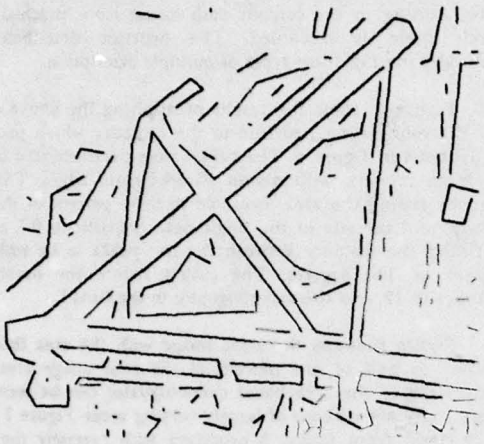


Fig. 3. Airfield edge elements.

We have so far experimented with three *producer* heuristics. Recall that a *producer* is supplied with the most recently linked edge element and possibly a direction for the contour being linked. The simplest heuristic we have used, called *nssm* simply checks whether there is a *super segment* which contains the previous edge element, and continues past it. If so, that succeeding edge is returned, with a score of two. The *continuation* heuristic tries to find an edge element which lies on the linear extension of the previous edge. For example if it is given the edge defining the fuselage forward of the starboard wing (see fig. 2) the *continuation* heuristic suggests the edge behind the wing. Thus *continuation* is useful for following straight edges which are broken by intrusions. There are a number of parameters which can be adjusted; the variation allowed between the angles of the two edge elements, the distance the proposed continuation edge can start from the first (normalized for the length of the first), and how close the beginning of the new segment must actually be to the straight line which continues the first edge element. The first two of these are necessary to compensate both for the digital approximations of where the edges are, and for errors introduced by the line finding stage where good edges have been slightly deflected by small disturbances at their ends, incorrectly linked to them. The *continuation* heuristic uses the hash tables described above. Multiple continuation edges which meet all the above requirements may be found. The longest one is returned, with a score between 1.0 and 2.0, dependent on its length. The last heuristic we have used, *get_nearby* looks for edge elements which start near the end of the last edge in the contour. There is a parameter which controls the range of angles allowable for these edges. If the direction of the contour is known, then the search is constrained to the appropriate side of the last edge. Candidates are scored between 0.5 and 1.5, depending on their length. A list of candidates is returned in decreasing order according to their scores.

The *reapers* we have used have all been concerned with the convexity of the contour so far generated. They check whether the proposed new edge element follows a previously established direction for the contour, and whether there is a possible path from the end of the new edge element to the start of the contour, which keeps the region so defined convex. All these tests are loose, in the sense that concavities caused by edges within parameter *slop* of being convex, are ignored.

Figure 4. shows ribbons fitted to 194 of the 232 contours produced by using the three producers *nssm*, *continuation* and *get_nearby*, in that order, and the *cullers* described above (the ribbon fitter failed on the remaining 38 contours, as they had only a single peak when histogrammed on angles of the edge elements). The parameter *n_nodes* was set at value 2 for this and all experiments described here. Thus only two nodes of the search tree were kept in the ordered list, and the search terminated after completely developing two nodes. The effect of this is that for most nodes, the first edge suggested by a heuristic in the *producers* list, is the one which was used in the final solution. Thus super segments tend to get followed, unless they violate the direction of the contour already established. The second choice is for continuations of edges to be followed wherever possible.

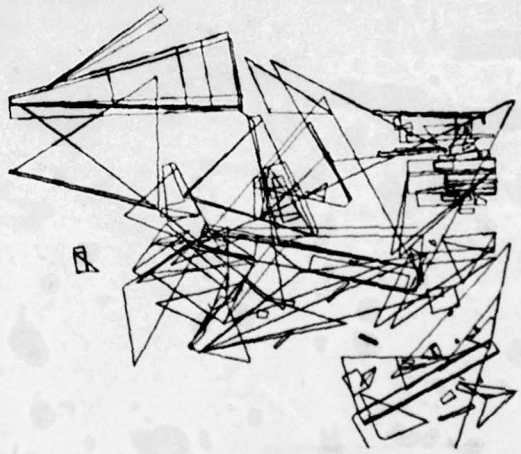


Fig. 4. All the ribbons.

It takes about thirty seconds CPU time on a KL-10 to produce all the contours for Figure 4, and approximate them with ribbons. It should be noted that the current implementation still carries a lot of developmental baggage, and has not yet been optimized. Clearly there are too many ribbons for a matching phase to do anything useful. We have used five culling heuristics to prune them. Setting the heuristic parameters gives control over the types of contours retained.

The simplest *culler* discards contours with less than a prescribed number of edges. In the experiments described here, we demanded at least three edges be linked together. To ensure that the contour bounds a region, we have a *culler* which calculates the exterior angles of the contour, and checks that their sum meets some threshold. Here we have used a threshold of 150 degrees. The next culler estimates the area of the region bounded by the contour, by filling in gaps between successive

edges with straight lines, and finding the area of the polygon. Again this is thresholded. The final two *cullers* rely on histogramming the edges of the contour, by their angles weighted for length. The first histograms modulo 180 degrees, finds the best peak and checks that it is greater than some threshold. This heuristic tends to catch contours which don't really bound a well defined region, but have managed to get past the exterior angle check. For instance, a contour made up of a wing tip, the trailing edge of the wing, the forward part of the opposite side of the fuselage, and the nose cone, passes the exterior angle check, but not the histogram check. Also if the threshold is greater than 0.5 (as in all the experiments described here) it tends to discard ribbons which are not longer than they are wide. The last heuristic histograms weighted angles for a full 360 degrees, and then checks the angular separation of the two best peaks. This can be used to ensure parallel sided ribbons. Another useful heuristic would be a length to width test. We have not used that in these experiments.

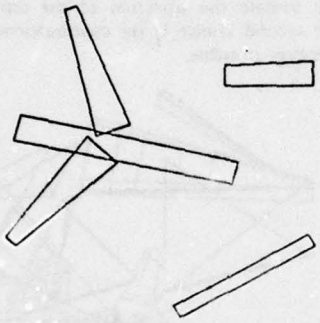


Fig. 5. Large regions.

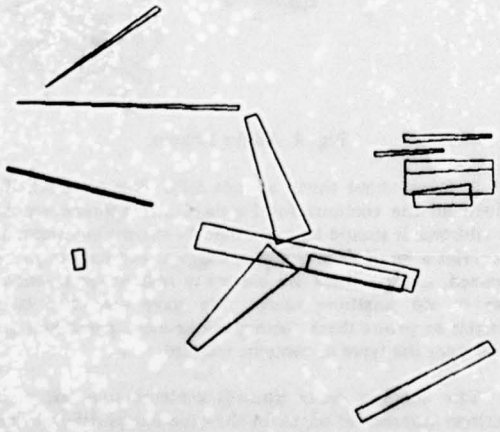


Fig. 6. Lower area threshold.

With the methods we have described one often gets multiple ribbons for a single region of the actual image. This occurs because many edges which appear in a single contour, are picked as root nodes for further tree searches, and so produce contours which are subsets of an already found larger contour. Nevatia [1974] also had this problem in fitting generalized cone descriptions to three dimensional laser range data. He discusses methods for choosing which description to use, based on the amount of shared boundary. So far the only disambiguator we have used is based on area comparison. It simply compares two ribbons (rather than contours) and if the center of gravity of each falls within the boundary of the other, then the ribbon corresponding to the contour with lowest score attached to its search node is discarded. The heuristic described here eliminates most of those types of multiple description.

Figure 5. show the results of applying the above *cullers* and disambiguating heuristic to the contours which produced the ribbons in figure 4. The cullers were parameterized to look for large regions, with almost parallel main edges. This was done by setting the area threshold to three percent of the total picture, and the size of the single peak heuristic to 0.7, and by confining the distance between the two peaks to be within 10 degrees of 180 degrees. The *cullers* reduce the number of contours to 12, and the disambiguator to the final 5.

Figure 6 shows the same image with the area threshold lowered to half of one percent of the total image area. The inadequacy of the area based disambiguator can be seen here, where there are ribbons of largely varying areas. Figure 7 shows the airfield from figure 3 processed with precisely the same parameters. The runways are picked out much better (see figure 8) when the continuation heuristic has its parameters loosened, as the line finder introduces large errors for some edges which actually correspond to a single broken straight line. A heuristic for finding anti-parallel lines to link would also improve performance.

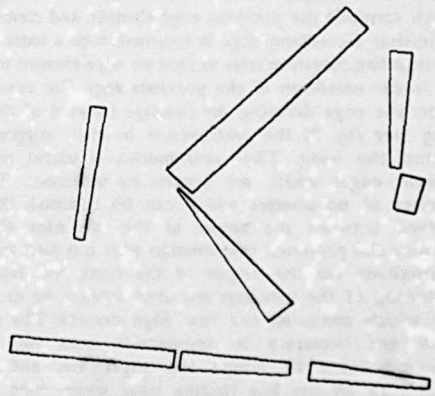


Fig. 7. Airfield.

Finally we give an example where finer work has been done looking for more detail in the airplane picture. We assume that the executive has found a tentative match for an airplane from the ribbons displayed in figure 5, and wants to look for rear stabilizers to confirm the match, and engine pods to identify the airplane type. The results of these two procedures is shown in figure 9 superimposed on the ribbons shown in figure 5. We manually played the role of the executive in this example.

Given the results of figure 5, the EM was set going once again, but the edges list of candidate root nodes was restricted to those starting near the rear of the fuselage. On the basis of the wing and fuselage areas, new ranges were set for the area of acceptable ribbons, and the condition on the parallelness of the main edges of ribbons relaxed because of the known shape of the stabilizers. The rest of the heuristics and parameters were unchanged.

For the engine pods, the area parameters were changed once more and the *continuation* heuristic was removed from the *producers* list. The search was restricted to the leading edges of the two wings and the rear of the fuselage. An automatic executive could use the results of such a search to help identify an airplane type. For this particular image, removing the *continuation* heuristic did not alter the results, however an automatic executive might decide that the engine pods are small and unlikely to have their edges interrupted by intrusions. Removing the heuristic reduces the possibility of making erroneous links to edges from nearby objects, such as trucks. It may seem strange that the small truck near to the starboard engine pod was not found in the engine search. It was considered but discarded by the single peak heuristic described earlier. If it had been elongated rather than square (or if we were looking for squarer engine pods) it would have been retained. Higher level processing in the executive would have then been responsible for deciding how to fit the candidate engine to the model.

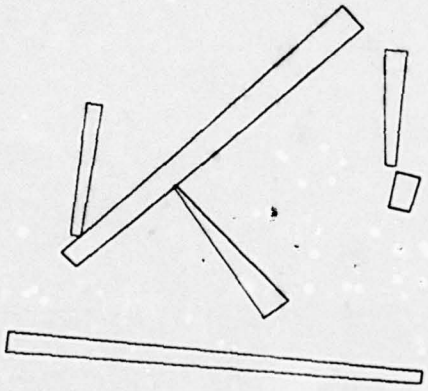


Fig. 8. Airfield with looser continuations.

Discussion

Although the experiments so far have been limited to a small number of images, we have found the algorithms and heuristics to be fairly stable over a wide range of parameter values. Often variations of 50 or 100 percent of a particular parameter produce only minimal changes in the ribbons identified. It thus seems promising that a reasoning system which is to select heuristics for the EM will be able to use fairly qualitative reasoning in deciding between a small number of values for each parameter. Furthermore, it should be possible to select heuristics which will be valid for a range of images for which only an approximate camera model (and thus ribbon size) is known.

Already the system seems powerful enough for special purpose executives to be written to identify certain classes of scenes. The ACRONYM project has more general goals however. The predictor and planner will be a general purpose executive, "programmed" by the generic and specific models of objects it is trying to locate in images. The predictor and planner will thus require knowledge of how volumetric elements and their relations will effect the type of heuristics which should be used for edge linking and region finding. It will need to know how to translate the explicit goals it can deduce from the models, into a choice of the correct heuristics and parameters. Knowledge of this sort will be incorporated into the rule set which it uses for reasoning (see Binford and Brooks [1979]). First, however, we will need to experiment further, to discover for ourselves how to better predict the heuristics to use, on the basis of what is expected in an image. Some of the reasoning which will be required was given during the explanation of the examples. This reasoning will have to be mechanized. Also we may find it necessary to increase the number of heuristics, from the current set. For instance an anti-parallel heuristic should improve the performance on runway and roadway type scenes.

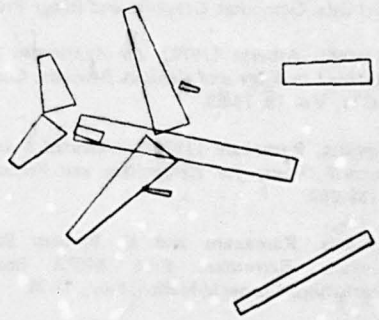


Fig. 9. An L1011.

It should also be noted that ACRONYM includes very powerful methods for disambiguation and identification in the matcher - using the symbolic descriptions provided by the Observability Graph. Thus, while the results presented in this paper seem very promising, the total system should be able to perform reasonably with an edge mapping module which doesn't meet our current expectations.

Acknowledgements

Prof. Nevatia, and Mr. Babu of USC have been extremely cooperative in providing us with the segment files used for this work. Tom Binford, Michael Lowry and Hans Moravec of Stanford have provided assistance and insights.

References

Ashkar, G. P. and J. W. Modestino [1978], *The contour Extraction Problem with Biomedical Applications*, Computer Graphics and Image Processing 7, 331-355.

Ballard, Dana H. and Jack Sklansky [1976], *A Ladder-Structured Decision Tree for Recognizing Tumors in Chest Radiographs*, IEEE Transactions on Computers, Vol. C-25, 503-513.

Binford, Thomas O. [1971], *Visual Perception by Computer*, Invited paper at IEEE Systems Science and Cybernetics Conference, Miami, Dec.

Binford, Thomas O, and Rodney A. Brooks [1979], *Geometric Reasoning in ACRONYM*, in these proceedings.

Brooks, Rodney A., Russell Greiner and Thomas O. Binford [1978a], *A Model-Based Vision System*, Proc ARPA Image Understanding Workshop, Cambridge, May., 36-44.

Brooks, Rodney A., Russell Greiner and Thomas O. Binford [1978b], *Progress Report on a Model-Based Vision System*, Proc ARPA Image Understanding Workshop, Carnegie-Mellon, Nov., 145-151.

Martelli, Alberto [1972], *Edge Detection Using Heuristic Search Methods*, Computer Graphics and Image Processing 1, 169-182.

Martelli, Alberto [1976], *An Application of Heuristic Search Methods to Edge and Contour Detection*, Communications of the ACM, Vol 19, 73-83.

Nevatia, Ramakant [1974], *Structured Descriptions of Complex Curved Objects for Recognition and Visual Memory*, Stanford AIM-250.

Nevatia, Ramakant and K. Ramesh Babu [1978], *Linear Feature Extraction*, Proc ARPA Image Understanding Workshop, Carnegie-Mellon, Nov., 73-78.

Sklansky, J. and H. Wechsler [1977], *Finding the Rib Cage in Chest Radiographs*, Pattern Recognition 9, 21-30.

HILL-SHADING AND THE REFLECTANCE MAP

Berthold K. P. Horn

Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract. Shaded overlays for maps give the user an immediate appreciation for the surface topography since they appeal to an important visual depth cue. A brief review of the history of manual methods is followed by a discussion of a number of methods that have been proposed for the automatic generation of shaded overlays. These techniques are compared using the reflectance map as a common representation for the dependence of tone or gray level on the orientation of surface elements.

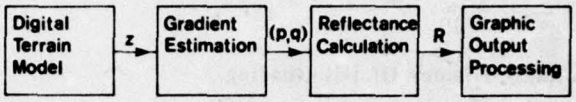


Figure 1: Block diagram of a system for the generation of relief shading. The grey-value is calculated by applying the reflectance map to the gradient estimate obtained by sampling neighboring points in the digital terrain model.

Introduction

Of several ways of depicting surface form on maps, hill-shading has the most immediate appeal and provides for quick comprehension of the topography. In this sense, hill-shading is complementary to the use of contours, which provide accurate terrain elevations but require careful scrutiny if one is to ascertain the surface form. Shaded maps are most important when the interpreter's time is limited, as in aviation, for users that are not trained cartographers and for small scale maps, where contours degenerate into messy tangles of lines.

Why then do we not see more shaded maps? One reason is the expense of present manual methods of production, which require skilled artists with good insight into cartography. Working from existing contour maps, ridge and stream lines extracted from such maps and at times aided also by aerial photography, they wield airbrushes in what is a slow, tedious and imprecise operation. Attempts at automation began with the notion that the gray levels used in the shading should derive from a model of how light might be reflected from a surface. Ignoring shadowing and mutual illumination effects, it seems clear that the reflected intensity will be a function of the local surface inclination. The choice of a method for calculating the gray tone based on the orientation of each surface element has however been the subject of occasionally bitter controversy for almost two centuries. Much of the difficulty stems from a lack of a common representation that would allow comparison of methods which appear at first glance to be incomparable.

The recently developed reflectance map constitutes such a common denominator. It is a simple device developed originally for work in machine vision where one is interested in calculating surface shape from the gray levels in an image. This is clearly just the inverse of the problem of producing shaded pictures from a surface model. The reflectance map is a plot of apparent brightness versus two variables, namely the slope of the surface element in the West-to-East direction and the slope in the South-to-North direction. Producing a shaded overlay for a map then is simply a matter of calculating these two slopes for each surface element and looking up the appropriate gray level in the reflectance map (see Fig. 1). This is a very simple, local computation that can be carried out efficiently even on enormous data-bases. The resulting gray levels can then be fed to a graphic output device that will produce a continuous tone or halftone photographic transparency from the given stream of numbers.

One important question that has to be settled is what reflectance map is to be used. Careful comparison of more than a dozen proposed shading methods shows that some of the simplest provide a good impression of the shape of the surface. These experiments also show that the most commonly used assumptions about surface reflectance do not lead to the best results, while simple monotonic functions of the surface slope in the direction away from the assumed light source work admirably. What matters is the visual impression, not theoretical rules [1]. One goal of this paper is a review of various hill-shading methods that have been proposed in the past in terms of their reflectance maps.

Early History Of Hill-Shading

Shading has been used in hand-drawn maps for many centuries. Leonardo da Vinci put it to good effect in his maps of Toscana, drawn in 1502 and 1503, that contained oblique views of relief forms illuminated from the left [1]. Wood-cuts of the area around Zurich in Switzerland drawn half a century later by Murer use shaded side-views as well. Overhead views using relief shading appear for the first time in maps of the same area drawn a century after that by Gygers, but these then gave way to less desirable forms [1].

The choice of the representation for relief forms depend to a great extent on the available reproduction technology. Woodcuts and engraving methods lend themselves to linear forms, where brightness of an area in the reproduction is controlled by the spacing and width of darkened lines. Useful directional, textural effects can be generated by orienting these line fragments, or hachures (Schraffuren), along lines of steepest descent. Crowding of such lines in steep areas may have given rise to notions of "steeper implies darker".

Lehmann proposed the first rigorous relationships [2,3] between surface slopes and quantities measurable on the printed map. In 1799, when his method (Böschungsschraffen) was published anonymously, the techniques for measuring the surface accurately at a large enough number of points did not exist. Results of this first method of illustrating shape are in some ways analogous to those one might obtain by illuminating a model of the surface from above, an arrangement that gives rise to images that are not easy to interpret.

Partly as a result of this, an alternate form (Schattenschraffen) evolved [4—6], in which the line thickness is varied according to the orientation of the local surface patch with respect to a light source usually assumed to be near the top left of the map, when it is oriented properly for viewing. For maps with North at the top this corresponds to North-West. Surface patches sloping down in that direction are portrayed with a light tone, while those sloping up in that direction get a dark tone. Since flat areas have no lines of descent, they remain white. This effect is unfortunate because it is the only departure from what would be seen if a diffusely reflecting model was illuminated obliquely. This makes such maps a little difficult to interpret. They are nevertheless superior to those made by the earlier method, as evidenced for example by the "Dufourkarte" of Switzerland made between 1842-1864 using this approach [1]. These methods for portraying surface shape preceded the widespread use of contours [7], in part because the latter require detailed surface measurements that were not available before the advent of photogrammetry.

While lithography was invented by Alois Senefelder in 1796, it found little application in cartography until around 1850. It permitted the production of multicolored maps, but more importantly, led to the use of halftones, destined to ultimately replace lines as a means of modulating the average reflectance in the printed map. W. H. Fox Talbot invented a photomechanical halftone process in 1852, but commerical success came only years after the patenting of the halftone screen by Frederick von Egloffstein in 1865, and the crossline screen of William A. Leggo in 1869.

Having access to these new reproduction schemes, Wiechel [8] developed shading methods (Schräglichtschummerung) to replace the use of hachures as described above. His fundamental paper, based in part on work by Burmester [9] on shaded pictures of *regular* surfaces, placed the field of hill-shading on a sound foundation. Wiechel discovered the error regarding flat surfaces, for example, and developed a graphic method for determining the gray value from contour interval and direction. Unfortunately, the means for controlled generation of halftones in response to surface orientation did not then exist and his work was ignored for a long time.

Hill-Shading In This Century

Two methods based on lines, this time contours instead of lines of steepest descent, were explored by Tanaka in the 1930's. His first method used the lines of intersection of the terrain surface with uniformly spaced, parallel, inclined planes [10,11]. Tanaka's initiative gave

rise to considerable discussion [12—19], partly in the form of an acrimonious debate [20—23]. His other method was based on portrayal of a terraced model of the terrain [24—26], an approach that had been used previously, unguided by his careful analysis [27—30]. While line-based methods give rise to beautiful, easy to interpret maps, they cannot show the fine detail of surface topography possible with halftones and must be based on smoothed, generalized information such as contours. These lines also tend to interfere with others used to portray planimetric information.

A shaded overlay can also be produced by photographing an appropriately illuminated scaled model of the surface. If this model has a matte or diffusely reflecting surface, a map overlay of high quality will result provided attention is paid to the projection geometry. While this was an approach taken early on [27], it really only became practical in the 1950's with the introduction of milling machines that allow an operator to carve a model by tracing contours on an existing map [31—38]. This is still an expensive, slow process however, in part because of the manual work required to smooth out the resulting "terraced" model.

The Swiss school of cartography improved on earlier forms [28,29,30,39—41] and developed shading to a fine art, producing numerous outstanding maps in this time [42—50]. Imhof argues that automated methods, such as relief model photography, cannot produce results nearly as impressive, since the cartographer cannot easily influence the process [1]. The manual shading method is however slow and expensive, and consequently has not been used except for small areas and those of particular interest or military importance. One cannot expect, with significant areas of the world still not mapped at large scales, and the rising cost of labor, that shaded overlays produced this way will be used in many maps.

Yoéli [51—59] saw the potential of the digital computer in dealing with this dilemma. It is possible to implement Wiechel's method based on oblique illumination of a diffusely reflecting surface if terrain elevations can be read into a computer and suitable continuous tone output devices are available. Yoéli was hampered by the lack of such devices at that time. Blachut and Marsik tried to simplify the required calculations to the point where a computer might not even be required [60,61]. Peucker helped popularize the whole idea of computer-based cartography [19,62—64], and found a piece-wise linear approximation to the equation for the brightness of a diffuse reflector that works well [63]. Many other interesting reports appeared during this time on the subject of hill-shading, too numerous to mention individually [65—72].

Brassel [73—77] took Imhof's admonitions seriously and tried to implement as much as had been formalized of the "Swiss manner". With the output devices available to him at that time it was not easy to judge whether the added complexity was worth the effort. All of these computer based methods require detailed digital terrain models. The storage capacity and techniques for handling this kind of information now exist [33,78—84]. as do the photographic output devices needed. There has been significant progress, too, in the automation of the generation of digital terrain models directly from aerial photographs [84—91], partly as a byproduct of work on orthophoto generation [92—95]. More compact and appropriate representations for these terrain models are under investigation [96—99], as are alternate methods for relief protrayal such as block diagrams [100—103].

Considerable progress has been made recently in the computer graphics area in the portrayal of regular objects with simple surfaces [104—115]. Early models for the reflection of light from matte surfaces [116—119] are being elaborated, including some for the material on the lunar surface [120—129]. In this context, work on models of the microstructure of surfaces is relevant [130—135]. In a recent effort in the machine vision area, a method was developed for portraying the dependence of brightness on surface orientation using the so-called reflectance map [136—140]. The reflectance map can be determined if the detailed geometric dependence of reflection from the surface [141,142] and the distribution of light sources are known. Alternatively, it can be found empirically, or derived directly by analyzing the interaction of lightrays with the surface microstructure.

As a result of the development of the reflectance map, the availability of detailed digital terrain data, small computers able to perform the simple calculations required and geometrically accurate gray-level output devices, we may say that automatic hill-shading has come of age.

Digital Terrain Models

For many applications of cartographic data it is useful to have machine-readable surface representations. Such terrain models are used for example in the design of roads and in order to determine the region irradiated by a radio frequency antenna. Initially, digital terrain models were generated manually by interpolation from existing contour maps. This is a tedious, error-prone process producing a digitized version of the surface

represented by the contours, which in turn is a smoothed, generalized version of the real surface.

The contour information on topographic maps is produced by manual scanning of stereo pairs of aerial photographs. Today, fortunately, stereo-comparators often come equipped with coordinate readouts that allow the extraction of information needed for the generation of digital terrain models [84]. Conveniently taken during orthophoto generation [92—95], the data tends to be accurate and detailed. Even more exciting is the prospect for machines that achieve stereo fusion without human help [84—91], since they will lead to the automatic production of digital terrain models. In the past such machines had difficulties dealing with uniform surfaces such as lakes, featureless surfaces, large slopes, and depth discontinuities, as well as broken surfaces, such as forest canopies. This is apparently still true when aerial photographs are used with disparities large enough to ensure high accuracy.

Various representations can be chosen for the surface elevation information. Series expansion, a weighted sum of mathematical functions such as polynomials, Gaussian hills or periodic functions may be used. These tend to be expensive to evaluate however and not accurate in approximating surfaces that have slope discontinuities. This is important for many types of terrain, at all but the largest scales. Perhaps the simplest surface representation is an array of elevations, $\{z_{ij}\}$, based on a fixed grid, usually square. Determining the height at a particular point is simple and the interchange of terrain models between users is easy since the format is so trivial. One disadvantage of this kind of surface representation is its high redundancy in areas where the surface is relatively smooth. The illustrations in this paper are based on digital terrain models consisting of arrays of elevation values.

Methods that achieve considerable data compression by covering the surface with panels stretched between specially chosen points have been developed [97—99]. These exploit the fact that real geographical surfaces are not arbitrary sets of elevations but have definite structure and regularity. Such representations may ultimately replace the simpler, more voluminous ones, if users can be persuaded to accept the greater programming complexities involved.

Digital terrain models may also be referred to as digital *elevation* models if they contain no information other than the elevation values.

The Reflectance Map

The human visual system has a remarkable ability to determine the distance to objects viewed, as well as their shape, using a variety of depth cues. One such cue is shading, the dependence of apparent brightness of a surface element on its orientation with respect to the light source(s) and the viewer. Without this particular depth cue we would be hard pressed to interpret pictures of smooth, opaque objects such as people, since other cues like stereo disparity and motion parallax are absent in a flat, still photograph. It can be shown that shading contains enough information to allow the observer to recover the shape. In fact, a computer program has been developed that can do this using a single digitized image [136].

Such work in the area of machine vision has led to a need to model the image-forming process more carefully [137]. The input to the visual-sensing system is image irradiance, which is proportional to scene radiance (here loosely called apparent brightness) [139]. Scene radiance in turn can be related to the underlying geometric dependence of reflectance of the surface material and the distribution of light sources [141,142]. Here we concentrate on the dependence of scene radiance on the orientation of the surface element. Shaded overlays for maps are interpreted by the viewer using the same mechanism normally employed to determine the shape of three-dimensional surfaces from the shading found in their images. Thus shaded overlays should be produced in a way that emulates the image-forming process, one in which brightness depends on surface orientation. This is why the reflectance map, which captures this dependence, is useful in this endeavor.

Consider a surface $z(x,y)$ viewed from a great distance above (see Fig. 2). Let the x-axis point to the East, the y-axis North and the z-axis straight up. The orientation of a surface element can be specified simply by giving its slope, p, in the x (West-to-East) direction and its slope, q, in the y (South-to-North) direction. The slopes p and q are the components of the gradient vector, (p,q). The apparent brightness of a surface element $R(p,q)$, depends on its orientation, or equivalently, the local gradient. It is convenient to illustrate this dependence by plotting contours of constant apparent brightness on a graph with axes p and q. This reflectance map [137] provides a graphic illustration of the dependence of apparent brightness on surface orientation. The pq-plane, in which the reflectance map is drawn, is called the gradient space, because each point in it corresponds to a particular gradient.

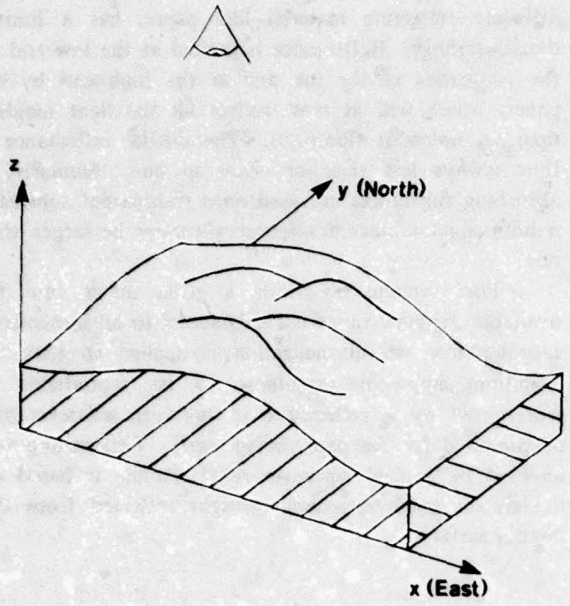


Figure 2: Coordinate system and viewing geometry. The viewer is at a great distance above the terrain so that the projection is orthographic.

Surface orientation has two degrees of freedom. We have chosen here to specify the orientation of a surface element by the two components of the gradient. Another useful way of specifying surface orientation is to find the intersection of the surface normal with the unit sphere. Each point on the surface of this Gaussian sphere again corresponds uniquely to a particular surface orientation. If the terrain is single-valued, with no overhangs, all surface normals will point more or less upwards and pierce the Gaussian sphere in a hemisphere lying above an equator corresponding to the horizontal plane. Gradient space happens to be the projection of this hemisphere from the center of the sphere onto a plane tangent at the upper pole.

While we will not use this representation in the calculation of relief shading, it is helpful in understanding previous attempts at graphical portrayal of the dependence of apparent brightness on surface orientation. The first such method was developed by Wiechel more than a century ago [8]. His brilliant analysis appears to have been largely ignored partly because it depended on mathematical manipulations that may have been inaccessible to many of the intended users. Later, Tanaka invented another method showing the variation of apparent brightness with surface gradient [10,11,24,25]. This second precursor of the reflectance map also appears to have found little following.

Position Dependent Effects

Since the reflectance map gives apparent brightness as a function of local surface gradient only, it does not take into account effects dependent on the *position* of the surface element. One such effect is illumination of one surface element by another. Fortunately this mutual illumination effect is small unless surface reflectance is quite high [137]. It is not known whether mutual illumination effects aid or hinder the perception of surface shape. They are difficult to calculate and so have not been emulated in work on hill-shading.

Another position dependent effect on apparent brightness is the blocking of light by one portion of the surface before it reaches another. Cast shadows can be calculated by determining which surface elements are not visible from the point of view of the light source [138]. Shadows cast by one complicated shape on another are hard to interpret however and apparently detract from the visual quality of shaded overlays [1,36,37]. They are thus rarely included.

Scattering of light by air molecules and aerosol particles changes the apparent brightness of a surface element viewed through the atmosphere. The brightness is shifted towards a background value equal to the brightness of an infinitely thick layer of air. The difference between the brightness and the background value decreases with the thickness of the gaseous layer through which the surface is viewed [143]. The resulting reduction in contrast as a function of distance is referred to as aerial perspective and can be a useful depth cue, although there is no general agreement that it aids the perception of surface shape. It has been used at times by map-makers and can be modeled easily [1,74,76,77]. The effect has not been added to any of the hill-shading schemes presented here in order to simplify comparisons.

Where Do Reflectance Maps Come From?

A reflectance map may be based on experimental data. One can mount a sample of the surface in question on a goniometer stage and measure its apparent brightness from a fixed viewpoint under fixed lighting conditions while varying its orientation. Instead, one can take a picture of a test object of known shape and calculate the orientation of the corresponding surface

element for each point in the image. The reflectance map is then obtained by reading off the measured brightness there.

Alternatively, one may use even more detailed information about light reflection from the surface. The bidirectional reflectance distribution function (BRDF) describes how bright a surface will appear viewed from one specified direction when illuminated from another specified direction [141,142]. By integrating over the given light source distribution one can calculate the reflectance map from this information [139]. Crudely speaking, the reflectance map is like a "convolution" of the BRDF and the source-radiance distribution.

Most commonly, reflectance maps are based on phenomenological models, rather than physical reality. The so called Lambertian surface, or perfect diffuser, for example, has the property that it appears equally bright from all viewing directions. It also reflects all light, absorbing none. It turns out that these two constraints are sufficient to determine uniquely the BRDF of such a surface, and from it the reflectance map, provided the positions of the light sources are also given. Some reflectance maps are based on mathematical models of the interaction of light with the surface. Such models tend to be either too complex to allow analytic solution or too simple to represent real surfaces effectively. Nevertheless some have come quite close to predicting the observed behavior of particular surfaces [133—135].

Here, new reflectance maps will be determined, based on proposed methods for producing shaded overlays for maps. Their derivation will not depend an understanding of the image-formation process or the physics of light reflection. Instead, they will require an analysis of how the brightness of a point in the overlay depends on the gradient of the underlying geographical surface.

Which reflectance map should be used? The answer to this question must depend on the quality of the impression a viewer gets of the shape of the surface portrayed. Various methods for producing shaded overlays can be compared by evaluating sample products and classified according to the corresponding reflectance maps. It will become apparent that in this way general conclusions can be drawn about a new method just by inspecting its reflectance map.

Normalization of Gray Tone

A picture made by applying varying amounts of lightabsorbing substances, such as ink, to an opaque, diffusely reflecting material like paper, has a limited dynamic range. Reflectance is limited at the low end by the properties of the ink and at the high end by the paper, which will at most reflect all the light incident upon it, unless it fluoresces. The diffuse reflectance is thus always less than or equal to one. Similarly, if absorbing substances are used on a transparent substrate, a limit applies, since transparency cannot be larger than one.

The problem of fitting a given image into the available dynamic range is fundamental to all methods of reproduction. A normalization is applied so that the maximum apparent brightness to be reproduced is represented by a reflectance of one (or whatever the maximum is for the paper being used). This scaling will have to be applied whenever relief shading is based on models of image-formation by light reflected from the terrain surface.

Gradient Estimation

The apparent brightness of a surface element depends on its orientation with respect to the viewer and the light source. The orientation of the surface element is described fully by a surface normal, or equivalently by the gradient. The components of the gradient are the slopes p (in the West-to-East direction) and q (in the South-to-North direction). These slopes have to be estimated from the array of terrain elevations. It is convenient to use a short-hand here for elevations in the neighborhood of a particular point (see Fig. 3). In the context of a single point at discrete coordinate (i,j), we will denote the elevation at that point by z_{oo}, while elevations of the adjacent grid points to the West and East will be called z_{-o} and z_{+o} respectively. Similarly, elevations at the points to the South and North will be denoted z_{o-} and z_{o+}.

$$\begin{array}{ccc} z_{-+} & z_{o+} & z_{++} \\ z_{-o} & z_{oo} & z_{+o} \\ z_{--} & z_{o-} & z_{+-} \end{array}$$

Figure 3: Short-hand notation for elevations of neighboring points.

The simplest estimates for the slope p might be

$$p_+ = (z_{+0} - z_{00})/\Delta x \text{ and } p_- = (z_{00} - z_{-0})/\Delta x,$$

where Δx is the grid interval in the West-to-East direction, expressed in the same units as the terrain elevations. These estimates are biased, actually estimating the slope half a grid interval to the right and left of the central point respectively. Their average however, the central difference, is unbiased,

$$p_c = (z_{+0} - z_{-0})/2\Delta x$$

Numerical analysis [144—146] teaches us that for certain classes of surfaces an even better estimate is obtained using a weighted average of three such central differences,

$$p_w = [(z_{++} + 2z_{+0} + z_{+-}) - (z_{-+} + 2z_{-0} + z_{--})]/8\Delta x$$

Symmetrically, one can estimate the South-to-North slope,

$$q_w = [(z_{++} + 2z_{0+} + z_{-+}) - (z_{+-} + 2z_{0-} + z_{--})]/8\Delta y$$

These expressions produce excellent estimates for the components of the gradient of the central point. The results depend on elevations in a 3×3 neighborhood, with individual elevation values weighted less than they are in the simpler expression for the central difference. This has the advantage that local errors in terrain elevation tend not to contribute as heavily to error in slope. At the same time, more calculations are required and three rows of the digital terrain model have to be available at one time.

Care has to be taken to avoid corruption of the slope estimates by quantization noise in the elevation values. Numerical problems due to the division of small integers may result when a terrain model is too finely interpolated, with limited vertical resolution. If it is necessary to generate many pixels in the output, it is better to interpolate the gray values produced by the shading algorithm.

Gradient Smoothing Effects

More complicated slope estimators than the ones described tend to introduce a *smoothing effect*, as can be seen by applying them near points of discontinuity in slope. To illustrate this more clearly, consider two horizontal smoothing operations H+ and H− that modify the terrain model as follows.

$$\text{H+: } z'_{00} = (z_{00} + z_{+0})/2$$

$$\text{and H−: } z'_{00} = (z_{-0} + z_{00})/2$$

It can now be seen that the central difference slope estimate, p_c, on the original terrain model, equals the biased estimate, p_+, calculated from the terrain model smoothed using H−, or, equivalently, the biased estimate, p_-, calculated from the terrain model smoothed using H+. Next consider two vertical smoothing operation V+ and V− in which the terrain model is modified as follows,

$$\text{V+: } z'_{00} = (z_{00} + z_{0+})/2$$

$$\text{and V−: } z'_{00} = (z_{0-} + z_{00})/2$$

The complicated slope estimate, p_w, can be shown to produce the same result as the first difference p_+ calculated from a terrain model smoothed by applying H−, V+ and V−. Similarly the slope estimate, q_w, equals q_+ calculated from a terrain model smoothed by applying V−, H+ and H− (Actually, since all of these operations are linear, their order can be arbitrarily rearranged). Perhaps any "smoothing" desired should be done as a separate editing operation, combined with the removal of "glitches" from the digital elevation model, rather than as part of the slope estimation. Also for terrain models of relatively limited size this smoothing may be undesirable. Some other slope estimators are simpler and introduce less smoothing. For example one can combine two biased estimates of the slope to get,

$$p_{1/2} = [(z_{++} + z_{+0}) - (z_{0+} + z_{00})]/2\Delta x$$

and symmetrically,

$$q_{1/2} = [(z_{++} + z_{0+}) - (z_{+0} + z_{00})]/2\Delta y$$

Here the average gradient in the top-right quadrant, $(z_{00}, z_{+0}, z_{++}, z_{0+})$, rather than at the central point is being estimated, using elevations in a 2×2 neighborhood only. For the graphic illustrations presented here, the

expressions for $p_{1/2}$ and $q_{1/2}$ were used to estimate the gradient.

At this time some terrain models are still produced by hand and have rather limited size. Rather than smoothing the terrain, one may wish to increase apparent resolution by some means. This can be done quite effectively by combining biased slope estimates (see Fig. 4). For every point in the terrain model, *four* gray values are produced corresponding to the four quadrants around it. Each is based on a different combination of the slope estimates (p_- or p_+) and (q_- or q_+) as appropriate for that quadrant. No miracles should be anticipated; this method cannot create information where there is none, but it can stretch what is available to its limits.

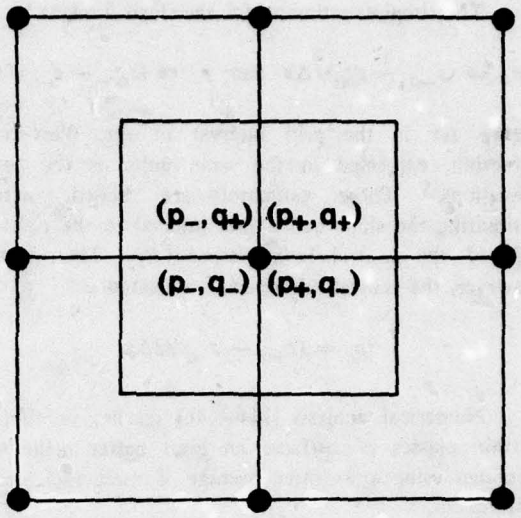


Figure 4: Combinations of biased slope estimates can be used to plot four times as many grey-tones as there are elevation values in the terrain model. The limited amount of data in a small terrain model may be stretched this way to produce reasonably detailed hill-shading output.

Rotated Gradients

It has been cartographic practice to assume a light source in the North-West at a 45° elevation above the horizon. It is helpful in this case to introduce a rotated coordinate system (see Fig. 5) with

$$p' = (p - q)/\sqrt{2} \quad \text{and} \quad q' = (p + q)/\sqrt{2}$$

If $\Delta x = \Delta y = \Delta$ say, then the slopes in the North-West to South-East and in the South-West to North-East direction, can be estimated particularly easily by combining the formulas for p_w and q_w,

$$p'_w = [(z_{+0} + z_{+-} + z_{0-}) - (z_{-0} + z_{-+} + z_{0+})]/4\sqrt{2}\,\Delta$$

$$q'_w = [(z_{0+} + z_{++} + z_{+0}) - (z_{0-} + z_{--} + z_{-0})]/4\sqrt{2}\,\Delta$$

If one wishes to estimate the slopes for the center of the top-right quadrant (in the unrotated coordinate system) rather than the central point one may combine the expressions for $p_{1/2}$ and $q_{1/2}$ to get the simple formulas,

$$p'_{1/2} = (z_{+0} - z_{0+})/\sqrt{2}\,\Delta$$

$$\text{and } q'_{1/2} = (z_{++} - z_{00})/\sqrt{2}\,\Delta$$

One advantage of the rotated coordinate system stems from the fact that models of surface reflectance considered here are symmetric with respect to a line pointing towards the source. That is, a surface element

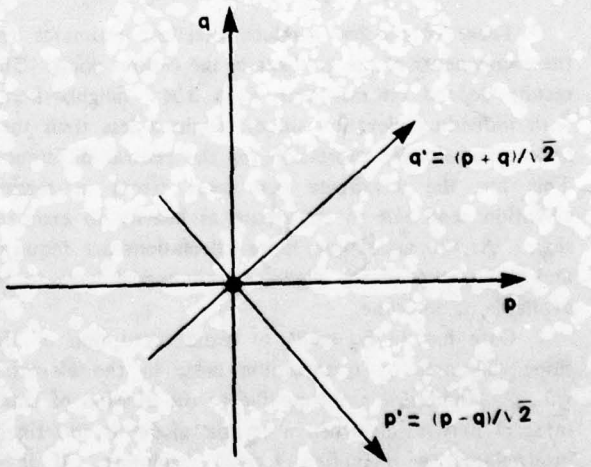


Figure 5: Rotated coordinate system that may be convenient when the assumed light-source is in the North-West. The reflectance map is symmetrical about the p'-axis.

with slopes $p' = p'_0$ and $q' = q'_0$ say, has the same apparent brightness as one with slopes $p' = p'_0$ and $q' = -q'_0$. Thus a lookup table based on the rotated coordinate system can be smaller, since only that half of the table corresponding to $q' > 0$ need be stored.

So far we have assumed that the grid of the terrain model is aligned with the geographical coordinates. If instead the whole model is rotated anti-clockwise by an angle θ, then slopes p'' and q'' can first be estimated from the model as described and then transformed as follows:

$$p = p'' \cos\theta - q'' \sin\theta$$

$$\text{and} \quad q = p'' \sin\theta + q'' \cos\theta$$

Alternatively, the model can be resampled to produce a new version on a grid aligned with the axes.

More complicated slope estimators than those discussed here do not seem called for, since the simple ones shown produce excellent results. Furthermore, estimators having wider support, while known to be more accurate for certain classes of functions such as polynomials, may perform worse on typical terrain with its discontinuities in slope along ridge and stream lines.

Exaggeration of Terrain Elevations

Compared to objects of a size that allow for easy manipulation by a human observer, the surface of the earth is in many places, though not everywhere, rather flat. The range of slopes is often so small as to cause disappointment with correctly proportioned models, so that height is often exaggerated in physical models. Similarly, shading based on models of light reflection from a surface tends to have undesirably low contrast. Here too terrain elevations may be exaggerated for all but the most mountainous regions. This is equivalent to multiplication of the components of the gradient by a constant factor, and corresponds to a simple transformation of the reflectance map. For reflectance maps based on reflection of light originating from an assumed source, a similar effect can often be achieved by a decrease in the elevation of the source. For flat surfaces the source may be lowered to a mere 10° or 20° above the horizon, where normally it might be at 45°.

Producing Shaded Overlays

The generation of shaded images from a digital terrain model using the reflectance map is straightforward. (see Fig. 1). For each point in the terrain model the local gradient (p,q) is found. The reflectance map then provides the appropriate brightness $R(p,q)$, to be plotted on a suitable gray-level output device. All computations are local and can be accomplished in a single pass through the image.

To illustrate these ideas a simple program is shown (see Fig. 6) that does not incorporate any of the elaborations described later on. Two arrays are used, Z, to store the terrain elevations, and B, to store the calculated brightness values. The latter has one row and one column fewer, since its entries correspond to points lying *between* those in the elevation array (the formulas for $p_{1/2}$ and $q_{1/2}$ are used). The spacing of the underlying

```
begin SHADING (N, M, DX, DY);
   array Z (N, M);
   array B (N-1, M-1);

         <read terrain elevations into array Z>

   do J = 1 to M-1;
      do I = 1 to N-1;
         B(I-1, J-1) := R( PE(I, J), QE(I, J));
      end do;
   end do;

         <write brightness values from array B>

   begin procedure PE(I, J);
      (Z(I, J) + Z(I-1, J) - Z(I, J-1) - Z(I-1, J-1)) / (2.0 *
DX);
   end PE;

   begin procedure QE(I, J);
      (Z(I, J) + Z(I, J-1) - Z(I-1, J) - Z(I-1, J-1)) / (2.0 *
DY);
   end QE;

   begin procedure R(P, Q);
      max(0.0, min(1.0, (1.0 + P - Q) / 2.0));
   end R;

end SHADING;
```

Figure 6: Simple program to generate shaded output from a terrain model.

grid is *DX* in the West-to-East direction and *DY* in the South-to-North direction. The procedures *PE(I,J)* and *QE(I,J)* estimate the slopes, while the procedure *R(P,Q)* calculates the corresponding brightness using a particularly simple reflectance map. The resulting values range from 0.0 (black) to 1.0 (white) and have to be scaled appropriately before they can be fed to a particular gray-level output device.

Typical terrain models are quite large and may exceed allowable array storage limits or even the address space of a computer. Fortunately only two (or three) rows of the terrain model are needed for the estimation of the local slopes. The program given can be easily modified to read the terrain model, and to write the calculated gray values, one line at a time. This makes it possible to deal with terrain models of essentially arbitrary size.

Next one should note that terrain models typically are stored using integer (fixed point) representation for elevations to achieve compactness and because elevations are only known with limited precision (an elevation may be given in meters as a 16-bit quantity for example). Similarly, gray values to be sent to a graphic output device are typically quantized to relatively few levels because of the limited ability of the human eye to discern small brightness differences and the limited ability of the device to accurately reproduce these (a typical output device may take values between 0 and 255.) The calculations can thus be carried out largely in integer (fixed point) arithmetic and even a simple computer is adequate.

Use Of Lookup Tables

Some of the formulas for reflectance maps discussed later on are quite elaborate and it would seem that a lot of computation is required to produce shaded output using them. Fortunately it is possible to make the amount of computation equally small in all cases by implementing the reflectance map as a lookup table. This table is computed only once, at the beginning.

Since elevations are quantized, so are the estimates of slope. It is therefore not necessary that one be able to determine the apparent brightness for all possible values of the gradient (p,q). Further, it is reasonable to place an upper limit on slope, so that only a finite number of possible values can occur (For example, if slopes between -1.55 and +1.60 are considered, in increments of 0.05, then there are only 64 possibilities for p and 64 for q, and a lookup table with 4096 entries can be used). A second justification for the use of a lookup table is the quantization of the gray values produced. It makes little sense to calculate the apparent brightness with very high precision only to coarsely quantize the result. A convenient rule of thumb is that the number of possible discrete values for each gradient component need not be more than the number of gray levels available from output device. The final choice of quantization must take into account both of the above considerations.

One can separate the estimation of slope from the calculation of gray value, and produce an intermediate file of coded surface gradient values. This file need not be larger that the original terrain model if the gradient is quantized properly (if p and q can each take on 64 values, each gradient can be encoded as a 12 bit value). Such a surface gradient file can be fed through a lookup table procedure to produce the final output. In this fashion different reflectance maps, encoded as different lookup tables, can be applied to a terrain model easily, with little more effort than reading and writing a file. The illustrations here were produced this way.

Many gray-level raster displays have a translation table between the image memory and the digital-to-analog converter driving the cathode ray tube intensity control. The quantized, packed reflectance map can be loaded into this lookup table, while the image memory is loaded with the coded slope matrix. This allows one to view the same terrain with a variety of assumed reflectance properties simply by reloading the translation table, which is small compared to the image memory.

Taxonomy of Reflectance Maps

Here we have discussed some of the issues one is likely to encounter when developing a program that produces shaded output. In the remainder of this paper we will analyze a number of proposed hill-shading methods in terms of their equivalent reflectance maps. Notational tools will be introduced as they are needed. Rather than proceed in strict historic order, we will discuss relief shading methods in the following groups:

1) *Rotationally symmetric reflectance maps* — gray tone depends on clope only.

2) Methods based on varying line spacing or thickness to modulate average reflectance.

3) Ideal diffuse reflectance and various approximations thereto.

4) Gray tone depends only on the slope of the surface in the direction away from the assumed light source.

5) Methods depending on more elaborate models of diffuse reflectance from porous material, such as that covering the lunar surface.

6) Models for gloss and lustrous reflection — smooth surface, extended source and rough surface, point source.

Average Reflectance Of Evenly Spaced Dark Lines

Some early methods for hill-shading achieve the desired control of gray tone by varying the spacing between printed lines. One advantage of this approach is the ease with which such information can be printed, since it is not necessary to first screen a continuous tone image. One disadvantage is the confusion created when the lines used for this purpose are layed on top of others portraying planimetric information. While the directional textural effects of the lines are important in conveying information about shape, we concentrate here on the average reflectance.

Consider inked lines with reflectance r_b covering an area of paper with reflectance r_w (see Fig. 7). The ratio of the area covered by ink to the area not covered is the same as the ratio of the width of the lines to the width of the uninked spaces. This in turn equals b/w, where b is the width of the inked line and w the width of the uninked space measured along any direction not parallel to the lines. If we ignore diffusion of light in the paper, then the average reflectance of the surface is

$$R = (w\, r_w + b\, r_b)/(w + b)$$

Or, $$R = r_w - b\,(r_w - r_b)/(w + b)$$

If, for example, the paper reflects all the incident light, and the ink none, then $r_w = 1$ and $r_b = 0$, so that $R = 1 - b/(w + b)$.

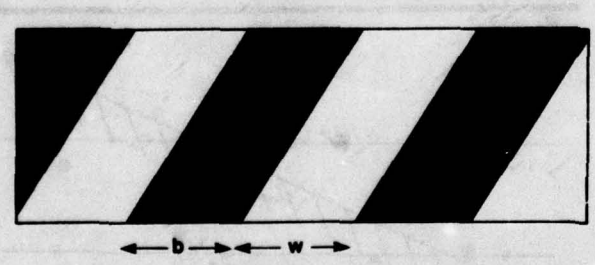


Figure 7: Magnified portion of surface covered with lines. The average tone depends on the fractional area covered by the lines, as well as the reflectance of the paper and the ink.

Slope Of The Surface In An Arbitrary Direction

In the calculation of gray value produced by some methods of hill-shading it is necessary to know the slope of the surface in an arbitrary direction, given the slope p in the West-to-East direction and the slope q in the South-to-North direction. Note that p and q are the first partial derivatives of the elevation z with respect to x and y respectively. Consider taking an infinitesimal step dx in the x direction and an infinitesimal step dy in the y direction. The change in elevation, dz, is given by

$$dz = p\, dx + q\, dy$$

Along a contour line for example, the elevation is constant, so that for a small step $dx = a\, ds$ and $dy = b\, ds$, we can write:

$$(p, q) \cdot (a, b)\, ds = 0$$

where "·" denotes the dot-product. The local direction of the contours, (a, b) is of course perpendicular to the local gradient (p, q).

Now consider taking a small step in an arbitrary direction, (p_o, q_o) say. That is let $dx = p_o\, ds$ and $dy = q_o\, ds$. The length of the step, measured in the xy-plane is,

$$\sqrt{p_o^2 + q_o^2}\; ds.$$

While the change in elevation is,

$$dz = (p_o\, p + q_o\, q)\, ds$$

Consequently the slope, change in elevation divided by length of the step, is,

$$s = (p_o\, p + q_o\, q)/\sqrt{p_o^2 + q_o^2}$$

If we let α be the angle between the vector (p_o, q_o) and the x-axis, then, the above can also be written,

$$s = p \cos \alpha + q \sin \alpha$$

The direction in the xy-plane in which the slope is maximal can be found by differentiating with respect to α. The direction of steepest ascent is (p, q) and the maximum slope equals $\sqrt{p^2 + q^2}$.

Lehmann's Böschungsschraffen A

One of the earliest methods for depicting surface shape using a form of shading is that of Lehmann [2,3]. Illustrations based on *ad hoc* scales of increasing darkness as a function of slope ("Schwärzegradscalen") had been published before, but there was no systematic analysis of this approach until the appearance of an anonymous publication attributed to Lehmann. In his method, short lines in the direction of steepest descent, called hachures, are drawn with spacing and thickness specified by rules that ensure that the fractional area darkened is proportional to the angle of inclination of the surface, θ. That is, steeper implies darker. The lines merge, producing a continuous black area, when θ exceeds some maximum value θ_o, typically 45° or 60°. The slope of the surface equals the tangent of the angle of inclination or "dip". Using the expression for the slope in the direction of steepest ascent, we get,

$$\tan \theta = \sqrt{p^2 + q^2}$$

Consequently, the average reflectance is,

$$R(p,q) = r_w - (r_w - r_b) \tan^{-1} \sqrt{p^2 + q^2} \,/\theta_o$$

When the angle of inclination exceeds the maximum, the lines coalesce and $R(p,q) = r_b$. We can also write the above in another form,

$$R'(\theta,\phi) = r_w - (r_w - r_b)\,(\theta/\theta_o)$$

Here, ϕ is the azimuth of the direction of steepest descent, a quantity that does not appear in the formula on the right, since apparent brightness here depends only on the magnitude of the slope. The direction *and* magnitude of the surface gradient can be found from a map prepared according to Lehmann's rules. The direction of steepest descent lies along the hachures, while the slope is directly related to the average tone that results from the width and spacing of these lines. In analyzing his method we have concentrated on calculating the average reflectance produced in the printed product. It should be pointed out that this method also gives rise to interesting textural effects that will not be discussed.

Another interesting aspect of Lehmann's method is that the lines or hachures were drawn starting on one contour and ending on the next. This greatly contributed to the later development of the contour representation (Isohypsen) for terrain surfaces, that was to ultimately replace most of these early attempts at portraying surface shape [7].

Contour Density B

Another method is based on the observation that lines on a contour map are more crowded in steep areas and that this crowding leads to darkening of tone or average gray value. In order to calculate the dependence of the average local reflectance on the gradient, (p, q), we have to determine the spacing of contour lines on the map. We assume that the surface is locally smooth and can be approximated by a plane, at least on the scale of the spacing between contour lines (If this is not the case, aliasing, or undersampling problems occur in any case).

Consider a portion of the surface with slope s in some direction not parallel to the contour lines (see Fig. 8). Assume that the map scale is k and the vertical contour interval δ. Then it is clear that the spacing between contours on the map, d, can be obtained from the formula for slope,

$$s = \delta/(d/k)$$

If we take the cross-section of the surface in the direction of steepest ascent, then $s = \sqrt{p^2 + q^2}$. As a

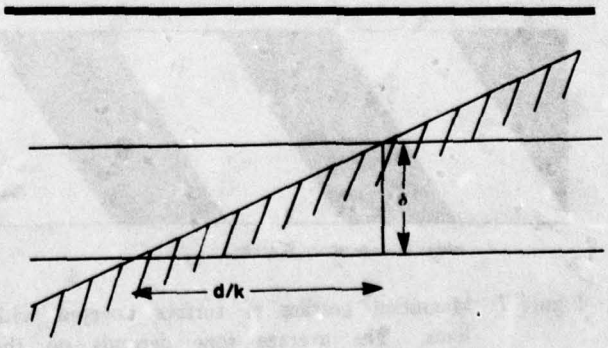


Figure 8: Spacing between successive contour lines along a given direction on the topographic map.

result we can write,

$$d = k\,\delta/\sqrt{p^2 + q^2}$$

On the map, $d = b + w$. That is, the spacing between contours is the sum of the width of the contour lines and the width of the blank spaces between them. The average reflectance then is,

$$R(p,q) = r_w - (b/k\delta)\,(r_w - r_b)\,\sqrt{p^2 + q^2}$$

The result can also be expressed as,

$$R'(\theta,\phi) = r_w - (b/k\delta)\,(r_w - r_b)\,\tan\,\theta$$

where θ is the inclination of the surface. The above expressions only hold if w is not negative. When the slope is too steep, contour lines overlap, and the average reflectance is simply equal to r_b. In the special case that $r_w = 1$ and $r_b = 0$, the above simplifies to,

$$R(p,q) = 1 - (b/k\delta)\,\sqrt{p^2 + q^2}$$

Typically $(b/k\delta)$ may equal 1 or $1/\sqrt{3}$.

Diffuse Surface under Vertical Illumination C D

The methods discussed so far produce tones that depend on the magnitude of the gradient only, not its direction. This is similar to the effect one would obtain if a physical model of the terrain was illuminated vertically, with the light source placed near the viewer. An ideal diffusing surface has an apparent brightness that is proportional to the cosine of the incident angle, i, as discussed later. This is the angle between the direction of the incident rays and the local normal, which, in the case of vertical illumination, is just θ. Therefore,

$$R'(\theta,\phi) = \cos\,\theta$$

Or,
$$R(p,q) = 1\,/\,\sqrt{1 + p^2 + q^2}$$

Instead of illumination from a point source, one may consider the effect of a distributed source. A uniform hemispherical source illuminating a diffusely reflecting surface leads to a result of the following form [139],

$$R'(\theta,\phi) = \cos^2(\theta/2) = \tfrac{1}{2} + \tfrac{1}{2}\cos\,\theta$$

Or,
$$R(p,q) = (1 + 1/\sqrt{1 + p^2 + q^2})\,/\,2$$

This reflectance map leads to flatter, even less interpretable pictures, since the range of reflectances has been halved and all reflectances have been shifted upwards by a half. In the derivation of the formula above, reflection from the surrounding terrain surface is ignored. If the terrain surface diffusely reflects a fraction ρ of the incident light, the constant term in the above expression is increased from ½ to $(1 + \rho)/2$, while the coefficient of cos θ decreases from ½ to $(1 - \rho)/2$. It is at times suggested that a component of surface brightness due to distributed illumination from the sky be added to that resulting from oblique illumination. This however typically detracts from the shaded result, rather than improving it.

The methods discussed so far give rise to rotationally symmetric reflectance maps, that can be described adequately by a single cross-section, showing tone versus slope [1,37]. This representation has sometimes been misused for asymmetric reflectance maps, where it does *not* apply. Rotationally symmetric reflectance maps produce shaded images that are difficult to interpret. Moving the assumed light source away from the overhead position gives rise to better shaded map overlays, but forces us to introduce some new concepts.

The Surface Normal

The surface normal is a vector perpendicular to the local tangent plane. The direction of the surface normal, **n**, can be found by taking the cross-product of any two vectors parallel to lines locally tangent to the surface (as long as they are not parallel to each other). We can find two such vectors by remembering that the change in elevation when one takes a small step dx in the x-direction is just $dz = p\,dx$, while the change in elevation corresponding to a step dy in the y-direction is $dz = q\,dy$. The two vectors, $(1,0,p)\,dx$ and $(0,1,q)\,dy$ are therefore parallel to lines tangent to the surface and so their cross-product is a surface normal.

$$\mathbf{n} = (1,0,p) \times (0,1,q) = (-p,-q,1).$$

Note that the gradient, (p,q), is just the (negative) projection of this vector on the xy-plane. A unit surface normal, N, can be obtained by dividing the vector **n** by its magnitude $n = \sqrt{1 + p^2 + q^2}$.

While it is convenient to specify directions as vectors, it is at times helpful to use spherical coordinates instead. A direction can then be given as an azimuth angle, ϕ, measured anti-clockwise from the x-axis, and a

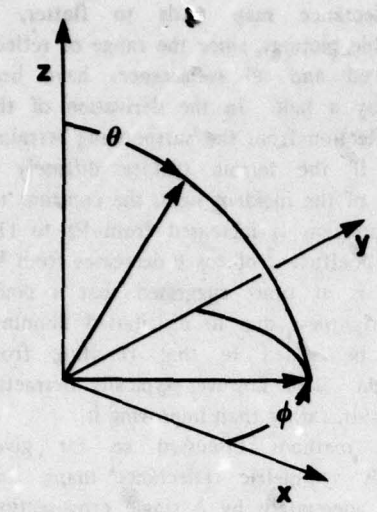


Figure 9: Definition of the azimuth angle, ϕ, and the zenith angle, θ. Here, azimuth is measured counter-clockwise from the x-axis in the xy-plane, while the zenith angle is measured from the z-axis.

polar or zenith angle, θ (see Fig. 9). (In navigation, the azimuth angle is usually measured clockwise from North, and the elevation angle is given instead of the zenith angle. These are just the complements of the angles used here.) The unit vector in the direction so defined equals,

$$N = (\cos\phi \sin\theta, \sin\phi \sin\theta, \cos\theta)$$

To find the azimuth and zenith angle of the surface normal we identify components of corresponding unit vectors. Then,

$$\sin\phi = -q/\sqrt{p^2+q^2}$$

$$\text{and} \quad \cos\phi = -p/\sqrt{p^2+q^2}$$

while,

$$\sin\theta = \sqrt{p^2+q^2} \,/\, \sqrt{1+p^2+q^2}$$

$$\text{and} \quad \cos\theta = 1 \,/\, \sqrt{1+p^2+q^2}$$

Conversely,

$$p = -\cos\phi \tan\theta \quad \text{and} \quad q = -\sin\phi \tan\theta$$

We will find it convenient to use both vector and spherical coordinate notation to specify direction.

Position Of The Light Source

The reflectance maps discussed so far are rotationally symmetric about the origin, only the magnitude of the gradient, not its direction affecting the resulting gray value. This corresponds to a situation where the light source is at the viewing position. Most hill-shading methods have the assumed light source in some other position, typically in the North-West, with a zenith angle of around 45° (θ_o = 45°, ϕ_o = 135°). The unit vector,

$$S = (\cos\phi_o \sin\theta_o, \sin\phi_o \sin\theta_o, \cos\theta_o)$$

points directly at the light source. A surface element will be illuminated maximally when the rays from the light source strike it perpendicularly, that is, when the surface normal points at the light source. By identifying components in the expression for the surface normal, $n_o = (-p_o, -q_o, 1)$, with those in the expression for the vector pointing at the source one finds that the components of the gradient of such a surface element are,

$$p_o = -\cos\phi_o \tan\theta_o \quad \text{and} \quad q_o = -\sin\phi_o \tan\theta_o$$

When the source is in the standard cartographic position, this means,

$$p_o = 1/\sqrt{2} \quad \text{and} \quad q_o = -1/\sqrt{2}$$

This standard position for the assumed light source was probably chosen because we are used to viewing objects lighted from that direction [1]. When we look at nearby objects in front of us, our body blocks the light arriving from behind us. Further, when writing on a horizontal surface, many of us find our right hand blocking light coming from that direction. We thus often arrange for light sources to be to the left, in front of us. While we can certainly interpret shading in pictures where the light source is not in this standard position, there seems to be a larger possibility of depth reversal in that case, particularly if the object has a complex, unfamiliar shape.

Returning now to the specification of the position of the light source, we find two identities that will be helpful later.

$$\cos(\phi-\phi_o) = (p_o p + q_o q) \,/\, [\, \sqrt{p^2+q^2}\ \sqrt{p_o{}^2+q_o{}^2}\,]$$

$$p_o p + q_o q = \tan\theta \tan\theta_o \cos(\phi-\phi_o)$$

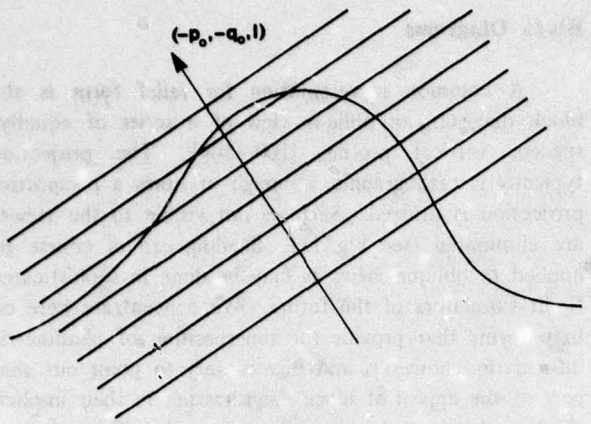


Figure 10: Side-view of a hill cut by inclined planes. Viewed from above, the lines of intersection crowd together where the surface slopes away from the equivalent source, while there are no lines where the terrain surface is parallel to the inclined planes.

It also follows that the slope of the surface in the direction, (p_0, q_0), away from the light source is,

$$s = \tan\theta \cos(\phi - \phi_0).$$

Tanaka's Orthographical Relief Method E

A method proposed by Tanaka in 1930 [10,11], involves drawing the lines of intersection of the surface with evenly spaced *inclined* planes. These planes are oriented so that their common normal points towards an equivalent light source (see Fig. 10). Thus slopes tilted away from this direction have contours spaced closely, giving rise to heavier shading than that on horizontal surfaces, while surfaces lying parallel to the inclined planes are lightest. As in Lehmann's method, some information may be conveyed by the directional texture of the contours. Here we concentrate on the average reflectance only.

A contour is the intersection of the terrain's surface $z = z(x,y)$ with a plane. The equation $z = z_0$ applies to a horizontal plane appropriate for ordinary contours. For "inclined contours" an inclined plane is used with an equation of the form

$$(-p_0,-q_0,1) \cdot (x,y,z) \;/\; \sqrt{1+p_0^2+q_0^2} = z'_0$$

The vector $(-p_0,-q_0,1)$ is perpendicular to the inclined planes. Ordinary contours represent the locus of the solution of $z(x,y) = z_0$, while inclined contours are the loci of solutions of the equation,

$$[-p_0x - q_0y + z(x,y)] \;/\; \sqrt{1+p_0^2+q_0^2} = z'_0$$

We can now apply our analysis of the contour density model to the modified surface, $z'(x,y)$, defined by the left hand side of this equation! All we need are the slopes of this new surface. Differentiating the above expression with respect to x and y, we get,

$$p' = (p - p_0) \;/\; \sqrt{1+p_0^2+q_0^2}$$
$$q' = (q - q_0) \;/\; \sqrt{1+q_0^2+q_0^2}$$

Finally then,

$$R(p,q) = r_w - (b/k\delta)\,(r_w - r_b)\,\frac{\sqrt{(p-p_0)^2 + (q-q_0)^2}}{\sqrt{1+p_0^2+q_0^2}}$$

We obtain the expression for contour density, derived earlier, when $p_0 = q_0 = 0$. Also, in the special case that $r_b = 0$, $r_w = 1$, $p_0 = 1/\sqrt{2}$, and $q_0 = -1/\sqrt{2}$,

$$R(p,q) = 1 - (b/k\delta)\,\sqrt{(p-1/\sqrt{2})^2 + (q+1/\sqrt{2})^2}\;\;/\sqrt{2}$$

It is sometimes useful to express the apparent brightness as a function of the azimuth ϕ and zenith angle θ of the surface normal. If we let ϕ_0 be the azimuth and θ_0 the zenith angle of the normal to the inclined planes, then the formula can be rewritten as follows,

$$R'(\theta,\phi) = r_w - (b/k\delta)\,(r_w - r_b)\cos\theta_0$$
$$\sqrt{\tan^2\theta - 2\tan\theta\tan\theta_0\cos(\phi-\phi_0) + \tan^2\theta_0}$$

When $\theta_0 = 45°$, $r_b = 0$ and $r_w = 1$, then, as Tanaka showed [10,11],

$$R'(\theta,\phi) = 1 - (b/k\delta)\,\frac{\sqrt{1 - \sin 2\theta\cos(\phi-\phi_0)}}{(\sqrt{2}\cos\theta)}$$

How does one choose the parameter $(b/k\delta)$? Tanaka felt that the shading produced by his method should match that seen on a surface covered with an ideal material called a perfect diffuser. The apparent brightness of such a surface varies with the cosine of the incident angle, between the surface normal and a vector pointing at the light source. He introduced a parameter called the line factor that is the ratio of the width of the inked line, b, to, $k\delta/\sin\theta_o$, the interval between inclined contours for a horizontal surface. The line factor is just,

$$(b/k\delta)\sqrt{p_o^2+q_o^2}\,/\,\sqrt{1+p_o^2+q_o^2}$$

Tanaka proposed varying the line width b in order to produce shading that matches that seen on a perfect diffuser, but realized the impracticality of this approach for all but polyhedral surfaces [10,11]. Resigned to using a fixed line width, he chose to optimize the line factor by considering the brightness distribution on a spherical cap extending to 45° slope. With the source at 45° elevation, the least deviation from the brightness distribution one would see if the surface was a perfect diffuser is obtained when the line factor equals 0.3608. Consequently, $(b/k\delta) = .3608\sqrt{2}$. Finally then,

$$R(p,q) = 1 - .3608\sqrt{(p-1/\sqrt{2})^2 + (q+1/\sqrt{2})^2}$$

It is unfortunate that this method later gave rise to some misunderstanding as well as a less rigorous hybridized form [15].

Block Diagrams

A common representation for relief form is the block diagram, an oblique view of a series of equally-spaced, vertical profiles [100—104]. The projection typically is orthographic, although at times a perspective projection is utilized. Surfaces not visible to the viewer are eliminated (see Fig. 11). Shading can of course be applied to oblique views as may be done in sophisticated flight simulators of the future. We concentrate here on map forms that provide for superposition of planimetric information however, and digress only to point out that part of the appeal of block diagrams lies in their implicit shading, due to the variation in the spacing of lines.

Following the discussion in the last section, it is clear that the equivalent light-source position is in the horizontal plane at right angles to the vertical cutting surfaces. The analysis in the previous section then applies directly. Things are a little more difficult if the result is to be expressed in terms of the coordinate system of the surface rather than one oriented with respect to the viewer.

We can analyze the shading apparent in block diagrams by calculating the spacing between lines as a function of the surface orientation. Let a local surface normal be $\mathbf{n} = (-p,-q,1)$. A series of parallel planes, with common normal $\mathbf{s}$, cuts the terrain surface. The intersections of these planes with the surface are viewed from a direction specified by the vector $\mathbf{p}$. It is assumed that the viewer is at a great distance so that the profiles are projected orthographically along lines parallel to $\mathbf{p}$ (see Fig. 12).

The line of intersection of one of the cutting planes with the local tangent plane will be parallel to the vector $\mathbf{n} \times \mathbf{s}$, since the line lies in both planes and is therefore perpendicular to the normals, $\mathbf{n}$ and $\mathbf{s}$. Now construct a plane through the line of intersection and the viewer.

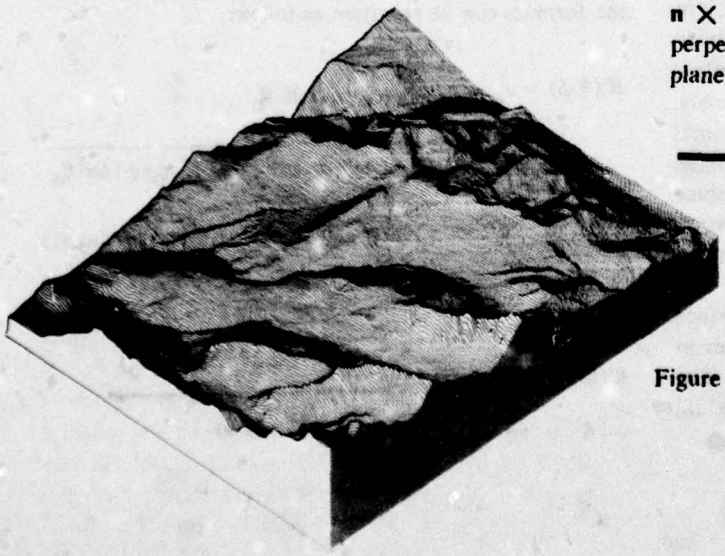


Figure 11: "Block-diagram" representation of terrain surface. This is an isometric projection of a series of uniformly spaced vertical profiles of the surface viewed from the South-East. Note the shading effect due to the variation in line spacing.

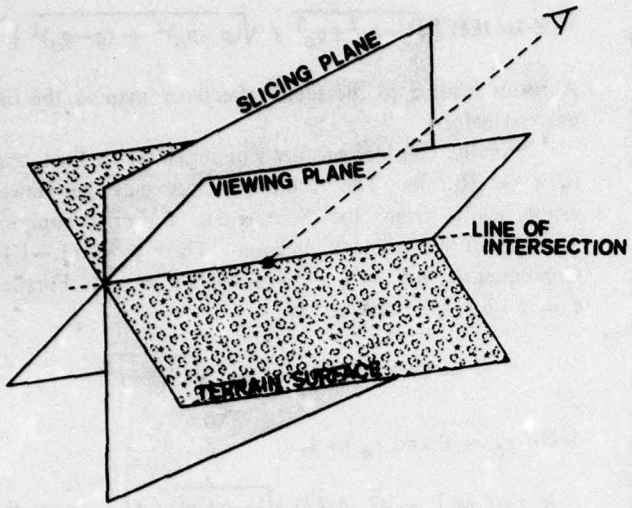


Figure 12: The viewing plane contains the viewer and the line of intersection of the slicing plane with the terrain surface. Line spacing in the block-diagram equals the spacing between successive viewing planes. The dotted line is parallel to the vector **p**.

This plane, called the viewing plane, contains both $\mathbf{n} \times \mathbf{s}$ and **p**. The normal **e** of the viewing plane must therefore be perpendicular to both and can be defined as,

$$\mathbf{e} = (\mathbf{n} \times \mathbf{s}) \times \mathbf{p}$$

or,
$$\mathbf{e} = (\mathbf{n} \cdot \mathbf{p})\ \mathbf{s} - (\mathbf{s} \cdot \mathbf{p})\ \mathbf{n}$$

If we let $\mathbf{v} = (x,y,z)$, then the equation for the local tangent plane can be written,

$$\mathbf{n} \cdot \mathbf{v} = c_n$$

for some value of the constant c_n. Similarly, the equation of a particular cutting plane is,

$$\mathbf{s} \cdot \mathbf{v} = c_s$$

Different values of c_s correspond to different cutting planes. The plane corresponding to the value $c_s + dc_s$ is separated from the plane corresponding to the value c_s by a distance dc_s/s, where s is the magnitude of the vector **s**. The equation for the viewing plane is just,

$$\mathbf{e} \cdot \mathbf{v} = c_e$$

Successive cutting planes will intersect the tangent plane in parallel lines. These give rise to parallel viewing planes corresponding to different values of the constant c_e. The spacing of these viewing planes is of interest, since it equals the spacing of the lines in the orthographic projection. The plane corresponding to the value $c_e + dc_e$ is separated from the plane corresponding to the value c_e by a distance of dc_e/e, where e is the magnitude of the vector **e**. In order to relate the spacing of lines in the block diagram to the spacing of the cutting planes we need to find the relationship between dc_e and dc_s.

A point **v** on the line of intersection lies in all three planes and therefore simultaneously satisfies the three equations given above for these planes. Expanding the last one of these, $\mathbf{e} \cdot \mathbf{v} = c_e$, we obtain,

$$(\mathbf{n} \cdot \mathbf{p})\ (\mathbf{s} \cdot \mathbf{v}) - (\mathbf{s} \cdot \mathbf{p})\ (\mathbf{n} \cdot \mathbf{v}) = c_e$$

or,
$$(\mathbf{n} \cdot \mathbf{p})\ c_s - (\mathbf{s} \cdot \mathbf{p})\ c_n = c_e$$

Here, c_n is fixed and so the relationship between changes in c_e and c_s is simply

$$dc_e = (\mathbf{n} \cdot \mathbf{p})\ dc_s$$

If the interval between cutting planes is δ and the map scale is k, then $dc_s/s = (k\delta)$. Consequently the spacing between lines in the block diagram, dc_e/e is,

$$d = k\delta\ (\mathbf{n} \cdot \mathbf{p})\ (s/e)$$

where e is the magnitude of the vector $\mathbf{e} = (\mathbf{n} \cdot \mathbf{p})\ \mathbf{s} - (\mathbf{s} \cdot \mathbf{p})\ \mathbf{n}$. Finally, we remember that

$$R(p,q) = r_w - (r_w - r_b)\ (b/d)$$

where b is the thickness of the lines. Thus,

$$R(p,q) = r_w - (b/k\delta)\ (r_w - r_b)\ (e/s)\ (1/\mathbf{n} \cdot \mathbf{p})$$

The view vector is tangent to the surface when $\mathbf{n} \cdot \mathbf{p} = 0$. When this dot product becomes negative, the surface is turned *away* from the viewer and should not be visible. Also note that $d = k\delta$, when $\mathbf{s} \cdot \mathbf{p} = 0$. One should therefore choose **s** and **p** so that they are not orthogonal, to avoid getting only evenly spaced parallel lines.

In the case of perspective projection, line density will increase with distance, and the resulting reflectance will be lowered because of a change in the effective scale factor k. If the projected profiles are plotted on a raster device, one has to also take into account the fact that

the number of dots per unit line length is not constant. The dot density varies as $\max[|\cos\theta|, |\sin\theta|]$, where θ is the angle between the line and the direction of the raster. This variation should be included if an accurate reflectance map is to be derived for output of this form.

Isometric Views of Vertical Profiles

The transformation between the terrain coordinate system and that of an observer viewing the terrain obliquely can be found by multiplying a rotation matrix corresponding to rotation by θ_p about the x-axis with a matrix corresponding to rotation by $(\pi/2+\phi_p)$ about the z-axis, where ϕ_p is the azimuth and θ_p is the zenith angle of the direction specified by the vector $\mathbf{p}$. If the coordinates in the observer's system are x', y', and z', one finds,

$$x' = -\sin\phi_p\, x + \cos\phi_p\, y$$

$$y' = -\cos\phi_p \cos\theta_p\, x - \sin\phi_p \cos\theta_p\, y + \sin\theta_p\, z$$

$$z' = +\cos\phi_p \sin\theta_p\, x + \sin\phi_p \sin\theta_p\, y + \cos\theta_p\, z$$

In the case of orthographic projection, the values of x' and y' are simply multiplied by the map scale k, to determine coordinates in the block diagram.

The general formula derived in the last section applies to all combinations of viewpoint and cutting plane orientation. It is interesting to look at a few special cases however. We can, for example, check our earlier result for the contour interval in an ordinary contour map. Here $\mathbf{n} = (-p,-q,1)$, as always, and $\mathbf{s} = (0,0,1)$, since we are considering the intersection of the surface with horizontal planes. Further, $\mathbf{p} = (0,0,1)$ since the viewer is vertically above the surface. Here then $s = 1$, $\mathbf{n}\cdot\mathbf{p} = 1$, and $\mathbf{e} = (p,q,0)$ The line interval is therefore,

$$d = (k\delta)/\sqrt{p^2+q^2}$$

The same reflectance map is obtained as before. Slightly more complicated is the case of Tanaka's inclined contours, where $\mathbf{s} = (-p_o,-q_o,1)$. Here, again, $\mathbf{n}\cdot\mathbf{p} = 1$, while,

$$\mathbf{e} = (p-p_o, q-q_o, 0) \quad \text{and} \quad s = \sqrt{1+p_o^2+q_o^2}$$

The line interval is therefore,

$$d = (k\delta)\left[\sqrt{1+p_o^2+q_o^2} \,/\, \sqrt{(p-p_o)^2 + (q-q_o)^2}\,\right]$$

A result leading to the same reflectance map as the one derived before.

Finally, consider profiles running West to East, that is, $\mathbf{s} = (0,1,0)$. The resulting traces may be viewed isometrically from the South-East, a fairly common arrangment for a block diagram. Then $\mathbf{p} = (1,-1,1)$. Consequently, $\mathbf{n}\cdot\mathbf{p} = (1-p+q)$ and $\mathbf{s}\cdot\mathbf{p} = -1$. Further, $\mathbf{e} = (-p,1-p,1)$ and hence,

$$d = (k\delta)\,(1-p+q) \,/\, (\sqrt{2}\,\sqrt{1-p+p^2}\,)$$

So, if $r_b = 0$ and $r_w = 1$,

$$R(p,q) = 1 - \sqrt{2}\,(b/k\delta)\,\sqrt{1-p+p^2} \,/\, (1 - p + q)$$

Similarly, for profiles running South to North, $\mathbf{s} = (1,0,0)$, and,

$$R(p,q) = 1 - \sqrt{2}\,(b/k\delta)\,\sqrt{1+q+q^2} \,/\, (1 - p + q)$$

At times two orthogonal sets of slicing planes will be used, producing a mesh on the surface. The reflectance map corresponding to this case can be found by adding the last two formulas and subtracting one from the result.

Wiechel's Contour-Terrace Model F

Imagine a three-dimensional model of the terrain built by stacking pieces of some material cut according to the shape of the contours on a topographic map [8]. If the thickness of the material is chosen correctly the model will be a scaled approximation of the terrain, looking a little like a tiered cake. Illuminating this construction with a distant point source will give rise to a form of shading since each contour "terrace" casts a shadow on the one beneath it (see Fig. 13). Wiechel [8] was the first to analyze the reflectance properties of such a surface. In order to calculate the average brightness of a portion of the model, when viewed from above, we must determine the width of the shadow relative to the width of the terrace.

The width of the shadow, measured perpendicular to the contours, varies, depending on the orientation of contours relative to the direction of the rays from the source. For example, when measured this way, the width is zero where the contour is locally parallel to the projection of the rays on the xy-plane. Measured in a

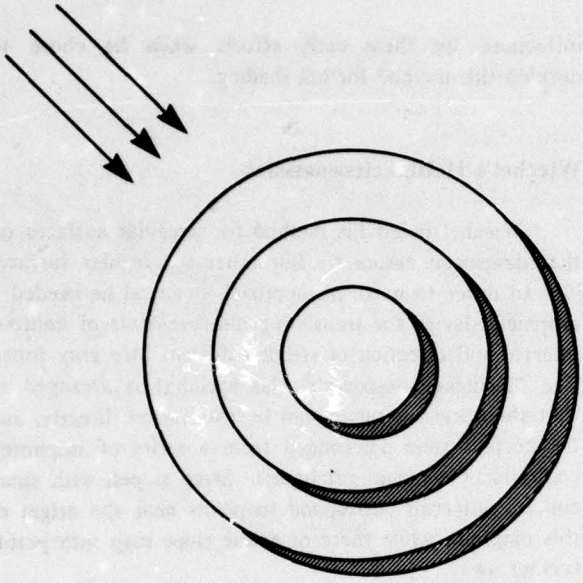


Figure 13: Shadows cast in the contour terrace model. The width of the shadows, measured perpendicular to the contours, varies with the direction of the contours relative to the direction of the incident rays.

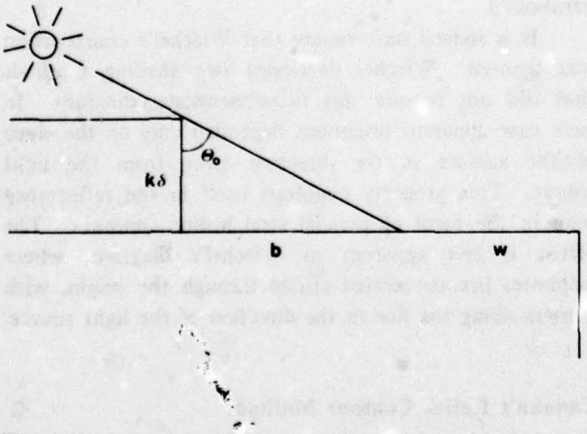


Figure 14: Section of the contour terrace model in a vertical plane containing the light-source. The width of the shadow, b, measured in this plane is constant, while the width of the terrace, $b + w$, depends on the slope of the surface in a direction parallel to the projection of the incident rays on the ground plane.

vertical plane containing the light source however, the width of the shadow is constant, since the terrace has a fixed height (see Fig. 14). If the light source has a zenith angle θ_o, the contour interval is δ, and the map scale k, then,

$$\tan \theta_o = (b/k\delta)$$

But,
$$\tan \theta_o = \sqrt{p_o^2 + q_o^2}$$

To calculate the average brightness we must know the width, d, of the terrace in the model, measured in the same vertical plane. The slope in this plane evidently is just

$$s = -k\delta/d$$

We know that the slope of a surface in the direction (p_o, q_o) is,

$$s = (p_o p + q_o q)/\sqrt{p_o^2 + q_o^2}$$

Solving for d from the last two equations and for b from the two before them, we get

$$b/d = -(p_o p + q_o q)$$

For example, when the local surface normal, $(-p,-q,1)$, is perpendicular to the direction to the source, $(-p_o,-q_o,1)$, their dot-product is zero and $b/d=1$. The terrace is then covered exactly by the shadow. In the above expression both the contour interval and the map scale have cancelled, as one might have predicted.

When $(p_o p + q_o q) < -1$, shadows coalesce and no further increase in b/d is possible. When, on the other hand, $(p_o p + q_o q) > 0$, the slope is facing *towards* the light source. This means that no shadow is cast. In this model, shading only occurs on slopes facing away from the source, while those facing towards it are all *uniformly* bright. This is certainly not what one would expect of a real surface and suggests that the contour-terrace model has some shortcomings. This is not surprising since apparent brightness depends on surface orientation, not height, and while the model represents height with reasonable accuracy it does a poor job of modeling surface orientation. Indeed the surface of the model is mostly horizontal, with some narrow strips of a vertical orientation. The latter are not even visible from above.

Wiechel noted that light would be reflected from these vertical surfaces onto the terraces [8]. The surface

thus appears brighter, viewed from above, near vertical surfaces facing towards the light source. He made the simplifying assumption that reflection produces *uniformly* bright patches with the same shape as shadows that would be cast were a source to be placed opposite the actual light source. This is not a reasonable assumption unless the vertical surfaces are made of narrow mirror facets, each oriented perpendicular to the direction of the incident light! In this case, surfaces illuminated by reflection as well as by direct light have a brightness twice that of those illuminated only by direct light. This version of the model is fortunately simple enough to be amenable to analysis. First note that, if we assume the surface to be an ideal diffuser, then the brightness of horizontal surfaces that are neither shadowed nor illuminated by reflection equals the cosine of the zenith angle of the source. Therefore, let $r_b = 0$ and $r_w = \cos \theta_o$, where,

$$\cos \theta_o = 1 \; / \; \sqrt{1+p_o^2+q_o^2}$$

$$R(p,q) = (1 + p_o p + q_o q) \; / \; \sqrt{1+p_o^2+q_o^2}$$

$$\text{Or, } R'(\theta,\phi) = [1 + \tan \theta \tan \theta_o \cos(\phi-\phi_o)] \cos \theta_o$$

When the source is in the standard position (North-West at 45°) this becomes,

$$R(p,q) = [1 + (p-q)/\sqrt{2}] \; / \; \sqrt{2}$$

Note that here apparent brightness already becomes equal to one when the angle of inclination is about 30.36° towards the light source. This may be contrasted with the case of the ideal diffuser, to be discussed later, where it reaches one only for an inclination of 45°. Wiechel used this model as the second approximation to the ideal diffuser (the first will be discussed later) and expressed his result as [8],

$$(\cos i \; / \; \cos e),$$

where i is the incident angle, and e is the emittance angle, here equal to θ. These angles will play an important role in the discussion of more recent methods later on.

According to Raisz and Imhof [1,28—30] terraced contour models were used in the late 1800's. An early example is an alpine excursion map published in 1865 that employed "contour shadows" [1]. The first attempts at photography of obliquely illuminated surfaces also used terraced terrain models [27]. Wiechel probably was influenced by these early efforts when he chose to develop this method for hill shading.

Wiechel's Helligkeitsmaassstab

Wiechel based his method for irregular surfaces on that developed earlier by Burmester for regular surfaces [9]. In order to make his approach practical he needed a graphical device for translating measurements of contour interval and direction of steepest descent into gray tones. The "Helligkeitsmaassstab" (his spelling) is arranged so that these measurements can be transferred directly, and the correct tone determined from a series of isophotes, contours of constant brightness. Steep slopes, with small contour intervals correspond to points near the origin of this diagram, while those of gentle slope map into points further away.

His diagram therefore is a sort of inside-out reflectance map. The main difference is that radial distance from the origin in gradient space is proportional to tan θ, while it is proportional to cot θ in this early precursor. This corresponds to a conformal mapping operation referred to as inversion with respect to the unit circle. Wiechel showed that his diagram corresponded to the image of an appropriately illuminated logarithmoid made of the desired material. The equation of this surface is $z = -\log \sqrt{x^2 + y^2}$. We saw earlier that the reflectance map can be thought of as the image of a paraboloid.

It is indeed unfortunate that Wiechel's construction was ignored. Wiechel developed two shading methods that did not require this two-dimensional diagram. In each case apparent brightness depended only on the slope of the surface in the direction away from the light source. This property manifests itself in the reflectance map in the form of parallel straight-line contours. The effect is less apparent in Wiechel's diagram, where isophotes become nested circles through the origin, with centers along the line in the direction of the light source.

Tanaka's Relief Contour Method G

Tanaka, in 1939, developed an ingenious method [24—26] for drawing the shadows one would see if one looked at a contour-terrace model. His method is based on the observation that the length of the shadow, measured in the direction of the incident rays, is constant. Using a pen with a wide nib one can trace the contours, while maintaining the orientation of the nib

parallel to the direction of the incident rays (as in roundhand writing). Only those portions of the contours are traced that correspond to slopes facing away from the assumed light source. Tanaka used black ink on gray paper for reasons that will become apparent. If the reflectance of this paper is r_g then,

$$R(p,q) = r_g + (r_g - r_b)\,(p_o p + q_o q)$$

provided $(p_o p + q_o q) < 0$, otherwise $R(p,q) = r_g$.

Tanaka also came up with a way of modulating the average reflectance of the paper in areas that corresponded to slopes facing *towards* the source. His approach is somewhat analogous to taking the negative of a picture of the contour-terrace model obtained by illuminating it from the other side. Thus white "shadows" are cast in the opposite direction to the black shadows. These can be drawn with white ink on gray paper using the same method as before except that now the section of the contours that correspond to slopes facing towards the light source are traced. It is easy to see that the resulting average reflectance will be,

$$R(p,q) = r_g - (r_g - r_w)\,(p_o p + q_o q)$$

where r_w is the reflectance of the white ink. When $(p_o p + q_o q) < 0$, no "shadows" appear and $R(p,q) = r_g$. Tanaka combined the two methods, tracing contours using both white and black ink. The corresponding reflectance map $R(p,q)$ equals one of the expressions above depending on whether the slope locally faces away from or towards the assumed source.

He apparently also experimented with nibs of different width for white and black ink. This corresponds to changing the elevation of the assumed sources. If the width of the nib is b, then the relationship is,

$$(b/k\delta) = \tan\theta_o = \sqrt{p_o^2 + q_o^2}$$

The results of this tedious manual method are most impressive [24—26]. One can write the above expressions in the alternate notation,

$$R'(\theta,\phi) = r_g + (r_g - r_b)\tan\theta\tan\theta_o\cos(\phi-\phi_o) \quad \text{when } \cos(\phi-\phi_o) < 0$$

$$R'(\theta,\phi) = r_g - (r_g - r_w)\tan\theta\tan\theta_o\cos(\phi-\phi_o) \quad \text{when } \cos(\phi-\phi_o) > 0$$

Tanaka preferred a reflectance for the gray background halfway between that of the black ink and the white ink. Placing the light source in the standard position we get,

$$R(p,q) = [1 + (p-q)/\sqrt{2}] \,/\, 2$$

Or, $$R'(\theta,\phi) = [1 + \tan\theta\cos(\phi-\phi_o)] \,/\, 2$$

This result can also be expressed as, $(\cos i \cos g) \,/\, \cos e$, where g is the phase angle, here equal to θ_o. Note that except for scaling by $\cos g$, this is the same result as that obtained by Wiechel for his contour-terrace model. One effect of this scaling is that apparent brightness rises to one only when the angle of inclination is 45°, on the other hand, horizontal surface now have a gray value of only 0.5.

Tanaka's Hemispherical Brightness Distribution

Tanaka needed a way to display the dependence of tone on surface orientation to permit comparison of the results produced by his two methods and what would be seen if the surface modeled were an ideal diffuser. He chose an oblique view of the brightness distribution on a spherical cap extending to 45° inclination [10,11,24—26]. If the cap is increased until it is a hemisphere, one obtains something like the reflectance map. One difference is that radial distance from the origin in gradient space is proportional to $\tan\theta$, while here it is proportional to $\sin\theta$. Thus, while the reflectance map is a central projection of the Gaussian sphere onto a horizontal plane, this is a *parallel* projection. Put another way: we are dealing here with an image of a hemisphere, while the reflectance map is the image of a paraboloid.

Tanaka's oblique views of the distribution of brightness versus surface orientation do not provide the quantitative information available in a contour representation such as Wiechel's. His method is nevertheless very helpful and it is unfortunate that few seem to have paid any attention to it, judging by the continued use of inappropriate forms. It is not uncommon for example to see the dependence of tone on surface orientation shown as a curve depending on one variable, slope, when it clearly depends on two, slope and the direction of steepest descent, or equivalently, the two components of the gradient.

Lambertian Surfaces

We now turn from graphical methods using variation in line spacing and line thickness to those utilizing continuous tone or halftone techniques. These are often based on a model of what the terrain would look like were it made of some ideal material, illuminated from a predetermined direction. The result differs from an aerial photograph, since no account is taken here of varying terrain cover, the light source is often placed in a position that is astronomically impossible, and the terrain model has been smoothed and generalized. Not being like an aerial photograph is an advantage, since aerial photographs, taken with the sun fairly high in the sky, often do not provide for easy (monocular) comprehension of surface topography.

The amount of light captured by a surface patch will depend on its inclination relative to the incident beam. As seen from the source the surface is foreshortened, its apparent (or projected) area equal to its true area multiplied by the cosine of the incident angle. Thus the irradiance is proportional to cos *i*. Strangely, it is commonly assumed that the *radiance* (apparent brightness) of the surface patch is also proportional to cos *i*. This is generally not the case since light may be reflected differently in different directions, as can be seen by considering a specularly reflecting material.

One can however postulate an *ideal* surface that reflects all light incident on it and appears equally bright from all viewing directions. Such a surface is called an ideal diffuser or Lambertian reflector and has the property that its radiance equals the irradiance divided by π [141,142]. In this special case the radiance is proportional to the cosine of the incident angle. No real surface behaves exactly like this, although pressed powders of highly transparent materials like barium sulfate and magnesium carbonate come close. Matte white paint, opal glass, and rough white paper are somewhat worse approximations, as is snow [130]. Most proposed schemes for automatic hill-shading are based on models of brightness distribution on ideally diffusing surfaces [8,10,11,51—59,74,76,77], even though there is no evidence that perception of surface shape is optimized by this choice of reflectance model. As we will see, reflectance calculations based on this model are not particularly simple either.

The cosine of the incident angle can be found by considering the appropriate spherical triangle (see Fig. 15) formed by the local normal, N, the direction towards the source, S, and the vertical, V. One then finds, as Wiechel already showed [8],

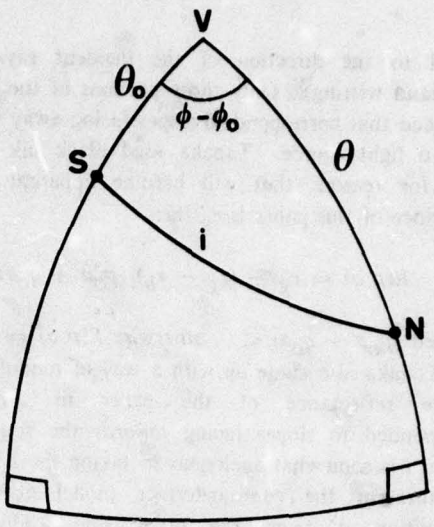


Figure 15: Spherical triangle used in calculating the incident angle, *i*, from the azimuth and elevation of the light-source and the azimuth and elevation of the surface normal. The direction towards the viewer is V, the direction to the source is S, while the surface normal is N.

$$R'(\theta,\phi) = \cos\theta_o \cos\theta + \sin\theta_o \sin\theta \cos(\phi-\phi_o)$$

Alternatively one can simply take the dot-product of the unit vector, N, normal to the surface and the unit vector, S, pointing towards the source [137,139].

$$\cos i = \frac{(-p,-q,1)\cdot(-p_o,-q_o,1)}{(\sqrt{1+p^2+q^2}\ \sqrt{1+p_o{}^2+q_o{}^2})}$$

The reflectance map (normalized so that its maximum is one) then is,

$$R(p,q) = (1+p_o p+q_o q) \,/\, (\sqrt{1+p^2+q^2}\ \sqrt{1+p_o{}^2+q_o{}^2})$$

When $(1+p_o p+q_o q) < 0$ the surface element is turned away from the source and is self-shadowed. In this case, $R(p,q) = 0$.

In the case of a point source of light at 45° zenith angle in the North-West, the reflectance map becomes

$$R(p,q) = [1 + (p-q)/\sqrt{2}] \,/\, [\sqrt{2}\,\sqrt{1+p^2+q^2}\,]$$

Peucker's Piecewise Linear Approximation

The computation of gray value using the equation for the cosine of the incident angle is complicated and slow because of the appearance of the square root. Peucker [63] experimented with a number of approximations that are easier to compute. He found that an adequate, piecewise linear approximation for slopes less than one, is

$$\begin{aligned} &.3441\ p - .5129\ q + .6599 \quad \text{for} \quad p+q > 0 \\ &.5129\ p - .3441\ q + .6599 \quad \text{for} \quad p+q < 0 \end{aligned}$$

or, $R(p,q) = .4285\ (p-q) - .0844\ |p+q| + .6599$

where $|p+q|$ denotes the absolute value of $(p+q)$. The above approximation produces excellent shaded overlays, that in fact seem easier to interpret that those produced using the exact equation for a perfectly diffusing surface.

Brassel's Adjustment Of Light Source Position

Perhaps the most outstanding examples of shaded maps come from Switzerland. Techniques for portraying the shape of the surface and integrating this information with planimetric detail have been perfected by a number of artists there [1,42—50]. The results of automated methods as described here, cannot compete with the beauty of their products. Nevertheless, automated methods do provide a systematic, accurate way for generating shaded overlays. They will become of particular importance when good digital terrain models become easily available. Brassel attempted to incorporate as much as possible of the Swiss manner into his program [73—77]. He quickly realized two problems with methods based purely on Lambertian reflectance models.

The first effect is explained as follows. Surface elements sloping away from the source are dark, while those tilted towards the source are brighter. Brightest are those that have the light rays falling perpendicularly on the surface. Surface elements sloped more steeply however, become *darker* again. This lack of monotonicity of brightness with slope is apparently disturbing and reduces the ability of the observer to interpret correctly the shape. Brassel ameliorated this effect by reducing the elevation of the light source in regions where this problem occurred.

If the zenith angle of the source, θ_o, is smaller than the zenith angle of the direction defined by the surface normal, θ, he moves the source to a new zenith angle, θ_n, that is a weighted average of θ_o and θ. To be precise,

$$\theta_n = \max\,[\theta_o,\ \alpha\theta + (1-\alpha)\theta_o]$$

where, $$\theta = \tan^{-1}\sqrt{p^2+q^2}$$

In his thesis [73], the weighting factor α was one, so that adjustment in elevation was complete. Curiously, this simple method has the effect of lowering the light source even for surface elements tilted *away* from the source, as long as the slope is large enough. The above method can also be expressed directly in terms of the components of the gradient. When $p^2+q^2 > p_o^2+q_o^2$,

$$\begin{aligned} p_n &= p_o\,(\sqrt{p^2+q^2}\ /\ \sqrt{p_o^2+q_o^2}) \\ q_n &= q_o\,(\sqrt{p^2+q^2}\ /\ \sqrt{p_o^2+q_o^2}) \end{aligned}$$

where p_n and q_n are the components of the gradient of a surface element oriented to be maximally illuminated by the adjusted light source. If there are no further adjustments of source position, the reflectance map in the specified region becomes,

$$R(p,q) = \frac{[1 + (p_o p + q_o q)\,(\sqrt{p^2+q^2}\ /\ \sqrt{p_o^2+q_o^2})]}{(1+p^2+q^2)}$$

Adjustment of the Azimuth of the Source

Next, Brassel observed that ridge and stream lines become indistinct when their direction was more or less aligned with a direction toward the source. Opposite faces of a mountain or valley may end up with similar gray values when the cosine of the incident angle is similar for the two, even though they have quite different surface orientations. Maximum contrast occurs when a linear feature lies at right angles to the direction of the incident light, and Brassel therefore moves the light source in azimuth towards the local direction of steepest ascent or descent (whichever is closer).

The amount of adjustment depends on two parameters (see Fig. 16). The maximum amount of adjustment is specified by w (55° for example), while the

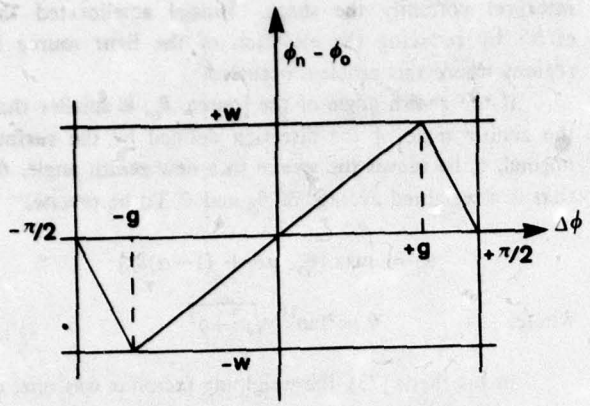


Figure 16: Sawtooth function giving adjustment of azimuth of the light source as a function of the angle between "regional" ridge and valley directions and the direction of the light source in Brassel's scheme.

azimuth difference at which this maximum occurs is specified by g (80° for example). The details of the computation are not very important but are given here for completeness. First, the azimuth of the direction of steepest ascent is computed using

$$\phi = \text{atan}(-q,-p)$$

where atan(y,x) is the direction of the line from the origin to the point (x,y) measured counter-clockwise from the x-axis. Next, the difference between ϕ and the azimuth of the source, ϕ_o, is reduced to the range $-\pi/2$ to $+\pi/2$ by adding or subtracting integer multiples of π. Let the result be $\Delta\phi$. The adjusted azimuth of the source is then calculated as follows,

$$\phi_n = \phi_o + w \ \text{sign}(\Delta\phi) \ \text{min}\,[|\Delta\phi|/g, (\pi/2-|\Delta\phi|)/(\pi/2-g)]$$

Where sign($\Delta\phi$) is +1 when $\Delta\phi > 0$, and −1 when $\Delta\phi < 0$. Now one can calculate the gradient (p_n, q_n) of the maximally illuminated surface element, or instead, use Wiechel's formula to get the cosine of the incident angle directly,

$$R'(\theta,\phi) = \cos\theta_n \cos\theta + \sin\theta_n \sin\theta \cos(\phi-\phi_n)$$

Here it should be pointed out that in Brassel's scheme the gradient, (p,q), used in the above formulas for adjusting the azimuth of the source is a *regional* value derived from ridge and stream lines in the area near a particular point. In this way the cartographer can influence the final appearance of the shaded overlay by altering these *manually* entered linear features. This method involves rather complicated global calculations that do not lend themselves to implementation in the straightforward way we have discussed. The apparent brightness of a surface element depends on both its orientation and some function of its surround.

A possible objection to this idea is that the distribution of light sources does not vary from place to place in a real imaging situation unless the sources are very close to the surface. It must be pointed out, however, that people seem to have little difficulty interpreting synthetic images where the assumed light source position varies. In fact, few notice such drastic changes in assumed light source position as are apparent in a recent map of the polar regions of Mars [147]. This may be related to the fact that our perception of shaded images does not give us a good appreciation for global differences in depth, instead giving us an excellent appreciation of local surface orientation patterns.

Whatever the merits of this argument, the above method can be modified to fit in with the notion of the *reflectance map, as defined earlier, if one uses the local* gradient (p,q) in the calculation of the adjusted source position. The illustration shown here uses this modified version. Note that in Brassel's scheme the adjustment in azimuth and zenith angle of the source are independent and can be carried out in either order.

Brassel also adjusted the apparent brightness according to the height of the terrain. This is a simple local computation that can be easily added to any of the basic methods presented here. It was not included here to simplify comparisons.

Alternate Light Source Adjustment Method K

Brassel used a piecewise linear adjustment in azimuth. A similar effect can be achieved using a smoothly varying function like

$$\sin\delta\phi = (\beta/2)\sin 2(\phi-\phi_o) = \beta \sin(\phi-\phi_o)\cos(\phi-\phi_o)$$

That is,

$$\sin\delta\phi = \beta\,[(p_o q - q_o p)(p_o p + q_o q)] \,/\, [(p^2+q^2)(p_o^2+q_o^2)]$$

Adjusting the azimuth of the source by $\delta\phi$ leads to a new position specified by,

$$p_s = p_o \cos \delta\phi - q_o \sin \delta\phi$$
$$q_s = p_o \sin \delta\phi + q_o \cos \delta\phi$$

Adjustment is complete for small angles when $\beta = 1$. The use of trigonometric functions is avoided in the above calculation, since both the sine and the cosine of $\delta\phi$ can be computed without them.

Next we turn to the adjustment in the elevation of the source. To avoid the peculiar phenomena of the lowering of the source even for surface elements turned *away* from it, we adjust the elevation according to the projection of the surface normal on a plane containing the source. When $p_s p+q_s q > p_s^2+q_s^2$,

$$p_n = p_s\,(p_s p+q_s q) \;/\; (p_s^2+q_s^2)$$
$$q_n = q_s\,(p_s p+q_s q) \;/\; (p_s^2+q_s^2)$$

In this region then the reflectance map becomes,

$$R(p,q) = \frac{\sqrt{1 + (p_s p+q_s q)^2/(p_s^2+q_s^2)}}{\sqrt{1+p^2+q^2}}$$

Otherwise it is calculated as before, that is, the cosine of the incident angle is

$$R(p,q) = (1+p_n p+q_n q) \;/\; (\sqrt{1+p^2+q^2}\;\;\sqrt{1+p_n^2+q_n^2})$$

The advantage of the above method of adjustment is that simple calculations in terms of the components of the gradient replace trigonometric equations in terms of azimuth and zenith angles.

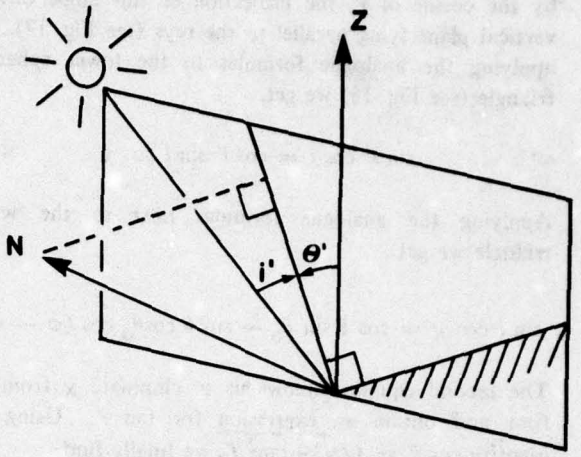


Figure 17: Projection of the surface normal on a vertical plane containing the assumed light-source. The projected normal is perpendicular to the line in which the plane cuts the terrain surface.

Wiechel's First Approximation — L M

The first serious analysis of an approach based on the shading seen on the surface of an obliquely illuminated matte object is that of Wiechel [8]. He started by assuming a perfectly diffusing surface and proposed connecting points of equal apparent brightness by isophotes. He correctly determined the brightness of a perfect diffuser as already mentioned. In order to make calculations less unwieldy he also suggested three approximations, the second of these being the contour-terrace model already discussed. His *first* method involved approximating the cosine of the incident angle, *i*,

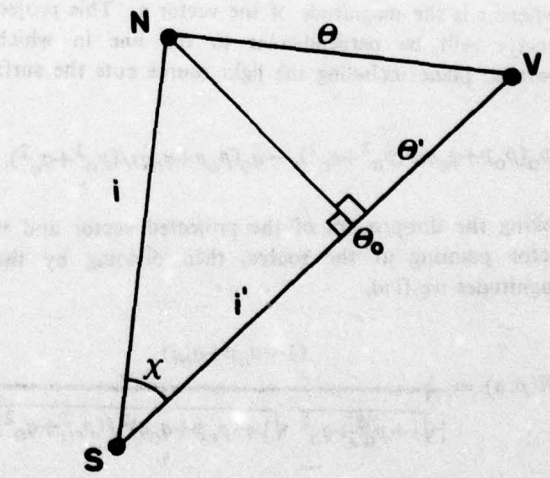


Figure 18: Spherical triangles used to calculate the projected incident angle, i', and the projected surface inclination, θ'. The direction towards the viewer is V, the direction to the source is S, while the surface normal is N.

by the cosine of i', the projection of this angle onto a vertical plane lying parallel to the rays (see Fig. 17). By applying the analogue formulas to the lower spherical triangle (see Fig. 18) we get,

$$\sin i' \cos i = \cos i' \sin i \cos \chi$$

Applying the analogue formulas next to the whole triangle we get

$$\sin i \cos \chi = \cos \theta \sin \theta_o - \sin \theta \cos\theta_o \cos (\phi - \phi_o)$$

The second equation allows us to eliminate χ from the first and obtain an expression for $\tan i'$. Using the identity $\cos i' = 1/\sqrt{1+\tan^2 i'}$, we finally find,

$$R'(\theta,\phi) = \cos i \;/\; [\cos \theta \sqrt{1 + \tan^2\theta \cos^2(\phi-\phi_o)}\,]$$

where, using the cosine formula as before,

$$\cos i = \cos \theta \cos \theta_o + \sin\theta \sin \theta_o \cos(\phi-\phi_o)$$

Alternatively one can project the vector $\mathbf{n} = (-p,-q,1)$ onto the plane with normal $\mathbf{s} = (q_o,-p_o,0)$. The result will equal,

$$\mathbf{n}' = \mathbf{n} - (\mathbf{n}\cdot\mathbf{s})\,\mathbf{s}/s^2$$

where s is the magnitude of the vector $\mathbf{s}$. This projected vector will be perpendicular to the line in which a vertical plane including the light source cuts the surface.

$$[-p_o(p_op+q_oq)/(p_o^2+q_o^2),\; -q_o(p_op+q_oq)/(p_o^2+q_o^2),\; 1]$$

Taking the dot-product of the projected vector and the vector pointing at the source, then dividing by their magnitudes we find,

$$R(p,q) = \frac{(1+p_op+q_oq)}{[\sqrt{1+p_o^2+q_o^2}\;\sqrt{1+(p_op+q_oq)^2/(p_o^2+q_o^2)}\,]}$$

This matches the expression for perfectly diffuse reflection for values of (p,q) along the line from the origin to the source point (p_o,q_o). When the source is in the standard position the equation becomes

$$R(p,q) = [1 + (p-q)/\sqrt{2}] \;/\; [\sqrt{2}\,\sqrt{1 + (p-q)^2/2}\,]$$

While these equations are more complicated than the original equations for the cosine of the incident angle, i, it must be pointed out that the angle i' can be estimated graphically by measuring the contour interval in a direction parallel to the incident light rays. The same is true of Wiechel's second approximation introduced earlier. This greatly simplifies the manual construction of shaded maps from contour maps, and makes it possible to use a simple one-dimensional scale for brightness instead of Wiechel's more elaborate "Helligkeitsmaassstab". This property manifests itself in the reflectance map by the appearance of parallel straight line contours. It is also interesting to note that Wiechel's "approximations" produces results that seem better than those obtained using the equation for the perfect diffuser. Unfortunately, experimentation at his time was limited because of the lack of appropriate technology for systematically generating continuous tone patterns. Apparently no maps made by this method were ever published [1].

Finally, Wiechel postulated a material that would *not* appear equally bright from all viewing directions, but instead had brightness varying as the cosine of the emittance angle. This was used in part to discuss the relationship between the contour-terrace model and the original surface, but also put forward as a third, "modified brightness" model that might be used in calculating gray tone. In this case brightness varies in proportion to $(\cos i \cos e)$. We can normalize his result here by dividing by the maximum of this product, $\cos^2(g/2)$, where g is the so-called phase angle, here equal to θ_o (The term phase angle stems from work on lunar photometry, where this angle equals the phase of the moon). Then,

$$R(p,q) = 2\,(\cos i \cos e) \;/\; (1 + \cos g)$$

$$= 2\,(1+p_op+q_oq) \;/\; [(1 + \sqrt{1+p_o^2+q_o^2}\,)\,(1+p^2+q^2)]$$

Incidentally, this function does not satisfy Helmholtz's reciprocity law [123], and therefore cannot correspond to the reflectance of any real surface illuminated by a point source.

Marsik's Automatic Relief Shading N

Blachut and Marsik further modified Wiechel's approximation, partly as a result of their dissatisfaction with the fact that a horizontal surface does not appear white when a perfectly diffusing material is assumed

[60,61]. This may have stemmed in part from early conventions in map-making where horizontal surfaces were portrayed without hachures [4—6]. Marsik also aimed for simpler calculations and considered the slope in the direction towards the source. For some reason, he proposed making the *density* of the printed result equal to the tangent of the projected slope angle θ' (see Fig. 17). Density is the logarithm (base 10) of the reciprocal of the reflectance. Applying the analogue rule to the upper spherical triangle (see Fig. 18) one can show that,

$$0 = \cos\theta \sin\theta' - \sin\theta \cos\theta' \cos(\phi - \phi_o)$$

Thus, $\tan\theta' = \tan\theta \cos(\phi - \phi_o)$,

and, $R'(\theta,\phi) = 10^{\tan\theta \cos(\phi - \phi_o)}$

Using the expression for the projected normal $\mathbf{n}'$ developed in the last section, or, remembering the expression for the slope in the direction (p_o, q_o), one can also show,

$$R(p,q) = 10^{(p_o p + q_o q)/\sqrt{p_o^2 + q_o^2}}$$

When $p_o p + q_o q > 0$, $R(p,q) > 1$ and so all surfaces facing towards the light source are white. No information is available to the viewer regarding surface shape in these areas. If the assumed light source is in the standard position we get the simple formula,

$$R(p,q) = 10^{(p-q)/\sqrt{2}}$$

Marsik also limited the density to a maximum of 0.7 to avoid interference with planimetric information on the map.

Lommel-Seeliger Law

Many surfaces have reflectance properties that differ greatly from those of an ideal diffuser. The photometry of rocky planets and satellites has intrigued astronomers for many years [120—126]. Several models have been proposed to explain the observed behavior. One of the earliest, developed by Lommel [118] and modified by Seeliger [119], is based on an analysis of primary scattering in a porous surface [125,127]. Their model consists of a random distribution of similar particles suspended in a transparent medium and results in a reflectance function that is given here in its simplest form,

$$1 \,/\, [1 + (\cos e / \cos i)]$$

unless $\cos i < 0$, when the surface is self shadowed. *Here i is the incident angle, and e is the emittance angle,* the angle between the local surface normal and the direction to the viewer, here equal to θ. The expression equals $1/(1 + \cos g)$ when $i = 0$, where g is the phase angle, here equal to θ_o. Using this value for normalization and remembering the expression for $\cos i$ one finds,

$$R'(\theta,\phi) = \frac{(1 + \cos\theta_o)}{\left[1 + \dfrac{\cos\theta}{(\cos\theta\cos\theta_o + \sin\theta\sin\theta_o\cos(\phi-\phi_o))}\right]}$$

$$R(p\,q) = \frac{\left[1 + 1/\sqrt{1+p_o^2+q_o^2}\,\right]}{\left[1 + \sqrt{1+p_o^2+q_o^2}\,/\,(1+p_o p+q_o q)\right]}$$

unless $(1+p_o p+q_o q) < 0$, *when* $R(p,q) = 0$. When the source is in the standard position,

$$(1+1/\sqrt{2})\,[1 + (p-q)/\sqrt{2}] \,/\, [(1+\sqrt{2}) + (p-q)/\sqrt{2}]$$

The Lommel-Seeliger law has been used in automated relief shading by Batson, Edwards and Eliason [72].

Based on detailed measurements and modeling, Fesenkov [122,126] and later Hapke [127—129] further improved the equations for the reflectance of the material in the maria of the moon. Hapke imagined the surface as an open porous network into which light can penetrate freely from any direction. His result has three components: the Lommel-Seeliger formula for reflection from a surface layer containing many scattering points of low reflectance, Schönberg's formula [121] for reflection from a Lambertian sphere and a complicated factor resulting from mutual obscuration of the particles. The results of such investigations are often expressed in terms of angles other than the ones introduced so far.

Luminance Longitude and Luminance Latitude

Another convention for specifying the orientation of the surface element relative to the direction of a light

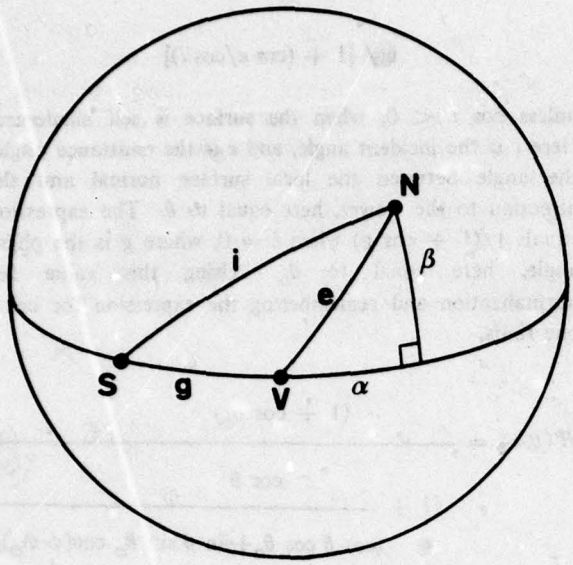


Figure 19: Luminance longitude α and luminance latitude β of a surface element are defined as the longitude and latitude of a patch on a sphere with the same orientation. Longitude and latitude are measured relative to the luminance equator through the light source S and the viewer V.

source and the viewer has become established in the work on planetary and lunar photometry. Imagine a sphere illuminated by a light source above the point S, viewed by an observer above the point V (see Fig. 19). These two points define a great circle that we take to be the equator. Then, points on the sphere can be referenced using the longitude, α, measured from the point V along the equator, and the latitude, β.

All possible surface orientations can be found on the sphere, and each surface orientation can be identified with some point, N say. The luminance longitude and luminance latitude corresponding to a particular surface orientation are the longitude and latitude of N. It is not difficult to show that,

$$\cos e = \cos\beta \cos\alpha \quad \text{and} \quad \cos i = \cos\beta \cos(\alpha+g)$$

$$\tan\alpha = (\cos e \cos g - \cos i) \,/\, (\cos e \sin g)$$

and,

$$\tan^2\beta = [1 + 2\,IEG - (I^2+E^2+G^2)] \,/\, [I^2 - 2\,IEG + E^2]$$

where we have used the shorthand notation, $I = \cos i$, $E = \cos e$, and $G = \cos g$. These results can also be expressed in terms of the components of the gradient:

$$\tan\alpha = (p_o p + q_o q) \,/\, \sqrt{p_o^2+q_o^2}$$

So $\tan\alpha$ is simply the slope in the direction away from the source. Now,

$$1 + 2\,IEG - (I^2+E^2+G^2) = (q_o p - p_o q)^2 \,/\, [(1+p^2+q^2)(1+p_o^2+q_o^2)]$$

$$I^2 - 2\,IEG + E^2 = [(p_o p+q_o q)^2 + (p_o^2+q_o^2)] \,/\, [(1+p^2+q^2)(1+p_o^2+q_o^2)]$$

The Lommel-Seeliger law can be expressed in terms of luminance longitude and luminance latitude as,

$$\cos(\alpha+g) \,/\, [\cos\alpha + \cos(\alpha+g)]$$

and it is clear from this form that scene radiance is independent of luminance *latitude*. This simplifies the problem of calculating the shape of the lunar surface from shading in a single image [136,137].

Minnaert's Reflectance Function P

Minnaert discusses a large variety of models for the reflection of light from rough surfaces [125]. He also proposed a class of simple functions of the form,

$$\cos^\kappa i \cos^{\kappa-1} e$$

intended to fit observations of the radiance of lunar material while obeying the reciprocity law [123]. Here κ is a parameter to be chosen so that the best fit with experimental data is obtained. This parameter is meant to lie between zero and one, with the above expression becoming equal to that for the perfect diffuser when $\kappa=1$. We can normalize this expression so it equals one when $i=0$,

$$R(p,q) = \cos^\kappa i \cos^{\kappa-1} e \,/\, \cos^{\kappa-1} g = [(1+p_o p+q_o q)/(1+p^2+q^2)]^\kappa \,(\sqrt{1+p^2+q^2} \,/\sqrt{1+p_o^2+q_o^2}\,)$$

Particularly Simple Reflectance Maps **Q R**

Several methods discussed here have reflectance depending only on the slope in the direction away from the assumed light source, leading to parallel straight line contours in the reflectance map. These include Wiechel's first and second "approximation", Tanaka's relief contour method, the "law" of Lommel and Seeliger, Minnaert's formula when $\kappa = \frac{1}{2}$, as well as Marsik's automatic relief shading. These methods are quite effective in producing overlays that are easy to interpret. One can construct more such reflectance maps, including some that are even easier to calculate. One possibility, for example, is,

$$R(p,q) = \frac{1}{2} + \frac{1}{2}\,(p' + a)/b$$

where, $\quad p' = (p_o p + q_o q) \,/\, \sqrt{p_o^2 + q_o^2}$

is the slope in the direction away from the source. Values less than or equal to zero correspond to black, while values greater than or equal to one correspond to white. The parameters a and b allow one to chose the gray value for horizontal surfaces and the rapidity with which the gray values changes with surface inclination. The simple program shown earlier (see Fig. 6) uses this form with $a=0$, $b=1/\sqrt{2}$ and $p_o=1/\sqrt{2}$, $q_o=-1/\sqrt{2}$.

A simple alternative, somewhat reminiscent of Lehmann's approach, is,

$$R(p,q) = \frac{1}{2} + (1/\pi)\,\tan^{-1}[(\pi/2)(p'+a)/b]$$

All possible slopes are mapped into the range from zero to one. This has the advantage that the reflectance does not saturate for any finite slope and all changes of inclination in the vertical plane including the source translate into changes in gray level.

Another way to achieve this effect is to use,

$$R(p,q) = \frac{1}{2} + \frac{1}{2}\,(p'+a) \,/\, \sqrt{b^2 + (p'+a)^2}$$

These three formulas are given in a form where the rate at which the gray value changes with surface inclination is the same at $(p'+a) = 0$.

Glossiness - The First Off-Specular Angle

Not all surfaces are matte. Some are perfectly specular or mirror-like. Since smooth, specularly reflecting surfaces form virtual images of the objects around them, patches of high brightness will appear when such a surface is illuminated by an extended source, like a fluorescent light fixture, or by light streaming in through a window. The size of the patches depends on the solid angle subtended by the source as well as the surface curvature, while the brightness distribution is that of the source.

To study reflection of an extended source in a specular surface, it is useful to introduce the "off-specular" angle, s, between the direction S to the center of the source and the direction S′, of the point that is specularly reflected to the viewer (see Fig. 20). This, incidentally, is also the angle between the direction to the viewer, V, and the direction, V′, in which the rays from the center of the source are specularly reflected.

We assume a circularly symmetric source, with brightness $L(s)$ at eccentricity s. This is the brightness the viewer observes in the specularly reflecting surface. Calculating the first off-specular angle s is simple using the appropriate spherical triangles.

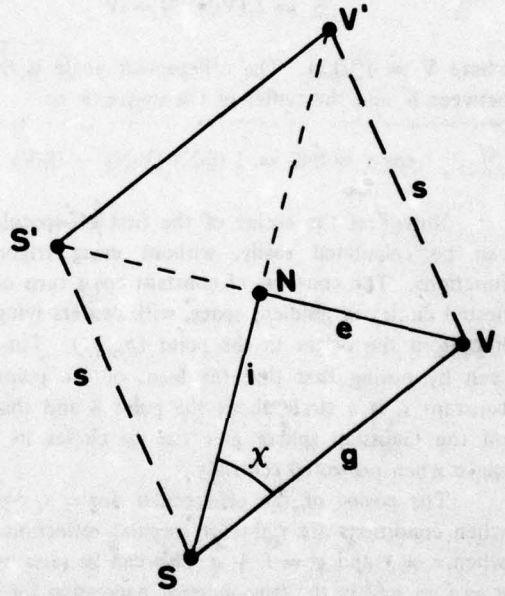


Figure 20: Spherical triangles used to calculate the first off-specular angle, s. It is the angle between S, the center of the source, and S′, the direction from which light is specularly reflected towards the viewer. Equivalently, it is the angle between V, the direction of the viewer, and V′, the direction in which light from the center of the source is specularly reflected.

$$\cos s = \cos 2i \cos g - \sin 2i \sin g \cos \chi$$
$$\cos e = \cos i \cos g - \sin i \sin g \cos \chi$$

Here, i is the incident angle, between the local normal and the direction to the source, $e = \theta$, is the emittance angle, between local normal and the direction to the viewer, while $g = \theta_o$ is the phase angle, between source and viewer. Eliminating χ from the two equations and expanding the sine and cosine of $2i$, one gets,

$$\cos s = 2 \cos i \cos e - \cos g$$

Substituting expressions in p and q for cos i, cos e and cos g one can rewrite this as,

$$[2\ (1+p_o p+q_o q)\ /\ (1+p^2+q^2) - 1]\ /\ \sqrt{1+p_o{}^2+q_o{}^2}$$

This result can also be obtained simply by finding the direction S′ from which a ray must come to be specularly reflected to the viewer V, by a surface element with normal N,

$$S' = 2\ (V \cdot N)\ N - V$$

where V = (0,0,1). The off-specular angle is the angle between S′ and the center of the source, S, so

$$\cos s = S \cdot S' = 2\ (S \cdot N)\ (V \cdot N) - (S \cdot V)$$

Note that the cosine of the first off-specular angle can be calculated easily, without using trigonometric functions. The contours of constant cos s turn out to be nested circles in gradient space, with centers lying on the line from the origin to the point (p_o, q_o). This can be seen by noting that that the locus of the point S′, for constant s, is a circle about the point S and that circles on the Gaussian sphere give rise to circles in gradient space when projected centrally.

The cosine of the off-specular angle, s, equals one when conditions are right for specular reflection, that is, when $e = i$ and $g = i + e$. This can be seen by setting $e = i = g/2$ in the trigonometric expression for cos s.

Bui-Tuong's Formula - Specular Surface, Extended Source

Having seen how to calculate the off-specular angle s, we can now make a reflectance map, by assigning the distribution of source brightness, $L(s)$. This function should be non-negative, monotonically decreasing with s, and equal to one when $s = 0$. For ease of calculation one choice might be

$$L(s) = \cos^n(s/2) = [\tfrac{1}{2}(1 + \cos s)]^{n/2}$$

where n is a number that defines how compact the bright patch is (A useful value might be around 20). So far, we have developed the reflectance map for a specular surface and a circularly symmetric source. Many surfaces, such as glazed pottery or smooth plastic, both glossy and difusse components reflection. Specular reflection takes place at the smooth interface between two materials of different refractive index, while the matte component results from scattering of light that penetrates some distance into the surface layer.

We can combine these two components as follows

$$R(p,q) = [(1-\alpha) + \alpha L(s)] \cos i\ /\ \cos(g/2)$$

where α determines how much of the incident light is reflected specularly. The expression is scaled so that its maximum is (approximately) equal to one. Here we have assumed the source, while distributed, is compact enough so that the diffuse reflection component can be approximated as cos i. The above expression obeys the reciprocity law of Helmholtz [123] which applies to real surfaces illuminated by a point source. Bui-Tuong used a reflectance function similar to the one derived above in his computer graphics work [112]. He apparently tried to model reflection from a surface that is not perfectly smooth. This requires a *different* off-specular angle however, as will be seen in the next section.

Luster - The Second Off-Specular Angle

Refulgency, gloss or shine can also appear when a point source is reflected in a surface that is not perfectly smooth. When a slightly uneven surface, of a material that gives rise to metallic or dielectric reflection, is illuminated by a point source, bright patches will be seen surrounding points where the local tangent plane is oriented correctly for specular reflection. The size of these patches will depend on the roughness of the surface and the surface curvature, while the distribution of brightness will depend to some extent on the texture of the microstructure of the surface.

In this case we will need to calculate the second off-specular angle, s', between the local normal, N, and the normal, N′, oriented for specular reflection of rays from the source S towards the viewer V (see Fig. 21).

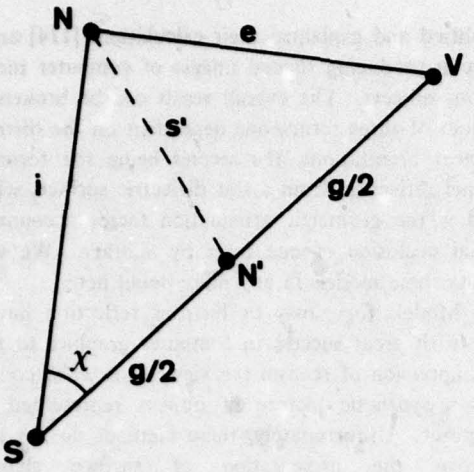


Figure 21: Spherical triangles used to calculate the second off-specular angle, s'. It is the angle between the actual surface normal, N, and a surface normal, N', oriented to specularly reflect rays from the source towards the viewer.

By considering the appropriate spherical triangles one finds,

$$\cos s' = \cos i \cos(g/2) - \sin i \sin(g/2) \cos \chi$$
$$\cos e = \cos i \cos g - \sin i \sin g \cos \chi$$

Eliminating χ from the two equations and expanding the sine and cosine of the phase angle g, one finds,

$$\cos s' = (\cos i + \cos e) \; / \; (2 \cos(g/2))$$

or, $$\cos s' = (\cos i + \cos e) \; / \; (\sqrt{2}\sqrt{1 + \cos g})$$

This result can also be obtained by finding the vector N', normal to a surface element oriented to specularly reflect a ray from the source in the direction of the viewer, V. That is,

$$N' = (S + V) \; / \; |S + V|$$

The off-specular angle is the angle between the actual surface normal N, and this vector N'.

$$\cos s' = N \cdot N' = [(S \cdot N) + (V \cdot N)] \; / \; \sqrt{2}\sqrt{1 + (S \cdot V)}$$

The surface microstructure of an uneven surface can be modeled by many randomly disposed mirror-like facets, too small to be optically resolved, each turned a little from the average local surface orientation. One can define a distribution, $P(s')$, describing what fraction of these microscopic facets are turned away from the average local normal by an angle s'. For ease of calculation one choice might be,

$$P(s') = \cos^n s'$$

Blinn's Formula - Rough Surface, Point Source T

One can use the fact that a normal, N', oriented for specular reflection of the point source towards the viewer, lies in the direction $(-p_1, -q_1, 1)$, where

$$p_1 = -\cos \phi_o \tan(\theta_o/2)$$
$$\text{and} \quad q_1 = -\sin \phi_o \tan(\theta_o/2)$$

We can also find N' by normalizing the vector (S + V), so that its third component equals 1.

$$p_1 = p_o \; / \; [1 + \sqrt{1+p_o^2+q_o^2}\,]$$
$$q_1 = q_o \; / \; [1 + \sqrt{1+p_o^2+q_o^2}\,]$$

A surface with gradient (p_1, q_1) is oriented just right to specularly reflect a ray from the source to the viewer. This can be seen by noting that when $p = p_1$ and $q = q_1$,

$$\cos i = \cos e = 1 \; / \; \sqrt{1+p_1^2+q_1^2}$$

and, $$\cos g = 2/(1+p_1^2+q_1^2) - 1$$

In any case,

$$\cos s' = (1+p_1 p+q_1 q) \; / \; (\sqrt{1+p^2+q^2}\,\sqrt{1+p_1^2+q_1^2}\,)$$

Note that s' will tend to be (roughly) half of s when both angles are small. Combining matte components of surface reflection with those from the rough outer surface we get,

$$R(p,q) = [(1-\alpha) + \alpha P(s')] \cos i \; / \; \cos(g/2)$$

The above reflectance map also obeys Helmholtz's reciprocity law and is normalized so that its maximum is (approximately) equal to one. Blinn and Newell give a similar reflectance function, claiming it was what Bui-Tuong had proposed [113]. The two are not the same

however since the two off-specular angles are different; in fact, the contours of constant s' are nested ellipses in gradient space, while, as mentioned earlier, the contours of constant s are nested circles. Indeed, Bui-Tuong's model corresponds to reflection of an extended, rotationally symmetric source in a specular surface, while the model presented in this section applies to reflection of a point source in a rough surface.

Blinn and Newell's Model for Specular Surfaces

One of the methods discribed by Blinn and Newell [113] assumes a perfectly specular surface in which the world surrounding the object is reflected. To make computations feasible, they imagine the surrounding objects at a distance great enough so that each part of the surround appears to lie in essentially the same direction from every point of the surface of the object. In this case one can imagine the brightness distribution of the surrounding objects projected onto the inside of a large sphere. The gray value used for a particular surface patch then is found by computing the direction S′ from which a ray must come to be specularly reflected to the viewer V, by a patch with surface normal N. We have already seen that,

$$S' = 2\,(V \cdot N)\,N - V$$

The appropriate gray value is then determined from the spherical distribution of brightness. In practice the sphere is mapped onto a plane by calculating the zenith angle, θ_o and azimuth, ϕ_o of S′ [113]. The brightness distribution can be equally well specified in gradient space [137], since it is also a projection of the Gaussian sphere.

Bouguer's Surface Model

Surface models incorporating randomly dispersed mirror-like facets were first studied around 1760 by Bouguer [117]. This type of micro-structure has been investigated extensively since then, despite the difficulties of reasoning about the three-dimensional nature of reflection from such surfaces. Recently, Torrance and Sparrow further elaborated on these models [133,134] in order to match more closely experimental data showing maximum brightness for angles of reflection *larger* than the incident angle. They included in their considerations the effects of obstruction of the incident and emergent rays by facets near the one reflecting the ray. Blinn simplified and explained their calculations [114] and used them in producing shaded images of computer models of various objects. The overall result can be broken into a product of three terms, one dependent on the distribution of facet orientations, the second being the formula for Fresnel reflection from a flat dielectric surface, while the third is the geometric attenuation factor accounting for partial occlusion of one facet by another. We will not discuss these models in any more detail here.

Models for glossy or lustrous reflection have been used with great success in computer graphics to increase the impression of realism the viewer has when confronted with a synthetic picture of objects represented in the computer. Unfortunately, these methods do not seem to improve the presentation of surface shape for cartographic purposes.

Colored Shading

It is often said that quantitative information about the surface cannot be obtained from relief shading [1]. Contour lines on the other hand do allow measurements of elevation and estimation of the gradient. Shading does provide *some* information about the gradient too, but cannot be used to determine both of its components locally, since only one measurement is available at each point. Since we can perceive the shape of objects portrayed by shaded pictures, it seems that these local constraints do lead to a *global* appreciation of shape, apparently based on our assumption that the surface is continuous and smooth.

If two shaded images, produced with the assumed light source in different positions, were available however, two measurements could be made at each point allowing one to determine the gradient *locally* [140]. It is inconvenient to work with two shaded overlays; fortunately though, they can be combined by printing them in different colors. In fact, yet another overlay can be added in a third color, but it adds no new information, since the two components of the gradient are already fully determined by the first two.

Colored shading corresponds to illumination by multiple sources, each of a different color. The exact color at each point in the printed result is uniquely related to the gradient at that point. Thus quantitative information *is* available in this new kind of map overlay. Further, ambiguities present in black and white presentations disappear. By positioning the light sources properly, one can avoid problems occasioned by the accidental alignment of ridge or stream lines with the

direction of incident light. Thus the need for *ad hoc* adjustments of the azimuth of the assumed light source is removed.

Colored shading is easy to interpret in terms of surface shape and effective in portraying surface form. It is unlikely however that it will be widely used because of the added expense of printing and conflict with existing uses of color in cartography to distinguish various kinds of planimetric information. Amongst other things, color is now used to code height and surface cover. Further, yellow is used in ordinary shading for sun-facing slopes, while violet is used for shaded regions [148]. This is thought to simulate the increased sky illumination component in areas turned away from the sun.

Summary and Conclusions

After a brief review of the history of hill-shading an efficient method for providing shaded overlays was described. It depends on a lookup table containing sampled values of the reflectance map. Traditional, manual methods were explored in terms of their equivalent reflectance maps, as were phenomenonlogical models used in the computer graphics community. Methods that have been proposed for mechanizing the generation of relief shading were also treated. The *automated method described here is very flexible, since it* can use any reflectance map.

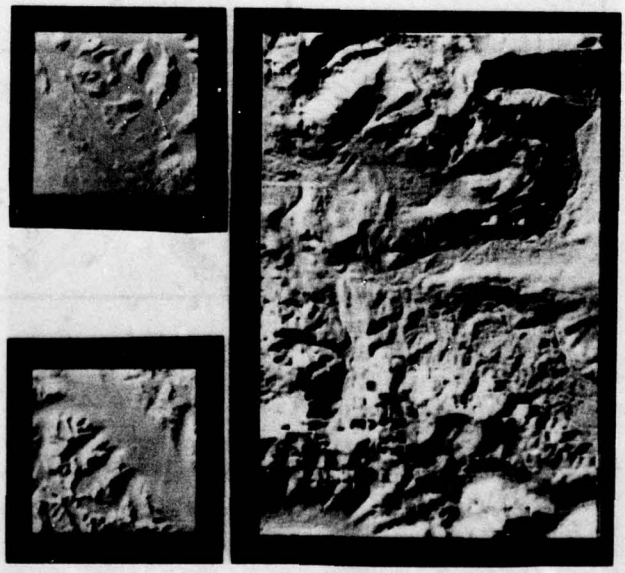
Some reflectance maps appear much better than others in conveying an immediate impression of surface shape. Rotationally symmetric reflectance maps, corresponding to overhead illumination of the terrain, are not very good for example. Perfectly diffuse reflectance is not optimal either. In fact, various approximations to the formula for a Lambertian reflector seem to produce better results. Simple monotonic functions of the slope in the direction away from the light source appear to be best. Glossy reflectance components, while very useful in the portrayal of regular objects, do not seem to be helpful in the case of complicated, irregular surfaces. Shading is an important depth cue. The choice of reflectance map should not be based on some *ad hoc* model of surface behavior, experimental measurement of reflectance of some material, or formulas that happen to be easy to calculate. Instead, one should use a reflectance map that gives rise to an immediate, accurate perception of surface shape.

It is important to arrange for the range of gray tones in the shaded overlays to be limited so as to avoid obscuring planimetric detail [149]. This is an area that has not received much attention so far. Another important issue relates to the appropriate scale for shaded overlays. Shaded overlays are useful for large scale maps. For small scale maps it is necessary to generalize the surface to avoid the appearance of complex textures that may be difficult to interpret [1,49,76,77,150]. This nonlinear process of removing small hills, ridges and valleys has not yet been satisfactorily automated.

An as yet unexplored possibility depends on finely sampled terrain elevations. This is the ability of shading to show fine detail. Contour maps have to be carefully generalized or smoothed to avoid showing confusing detail on a scale smaller than the contour interval. This is not the case with shading, although historically the manually produced maps have always shown only quite coarse features. We do not yet know whether the textures produced by the shading method when working from really fine terrain models will be confusing, or of great value in identifying different types of terrain.

Acknowledgments

I would like to thank Kurt Brassel, Thomas Peucker, George Lukes and Robert McEwen for generously supplying digital terrain model. Blenda Horn helped in the preparation of the text and Karen Prendergast created the figures. Helpful comments were provided by Robert Sjoberg, Katsushi Ikeuchi, and William Silver. Encouragment by Thomas Peucker, Kitiro Tanaka, and Kurt Brassel was instrumental in the generation of this paper.



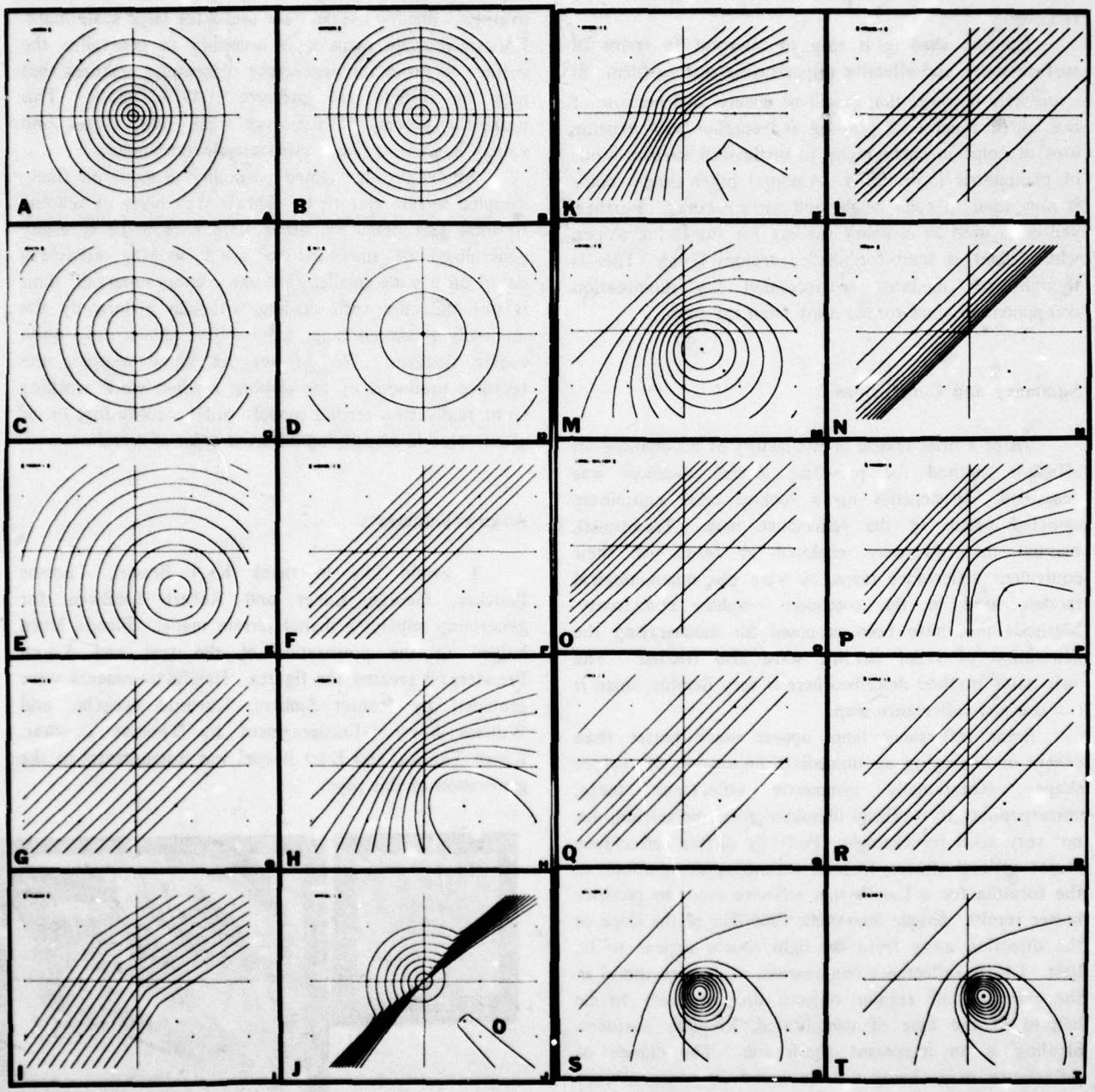


Figure 22: Reflectance maps in the order in which they were introduced in this paper. The first ten appear on the left, the last ten on the right. The letter codes correspond to the letters after the section headings of the corresponding sections.

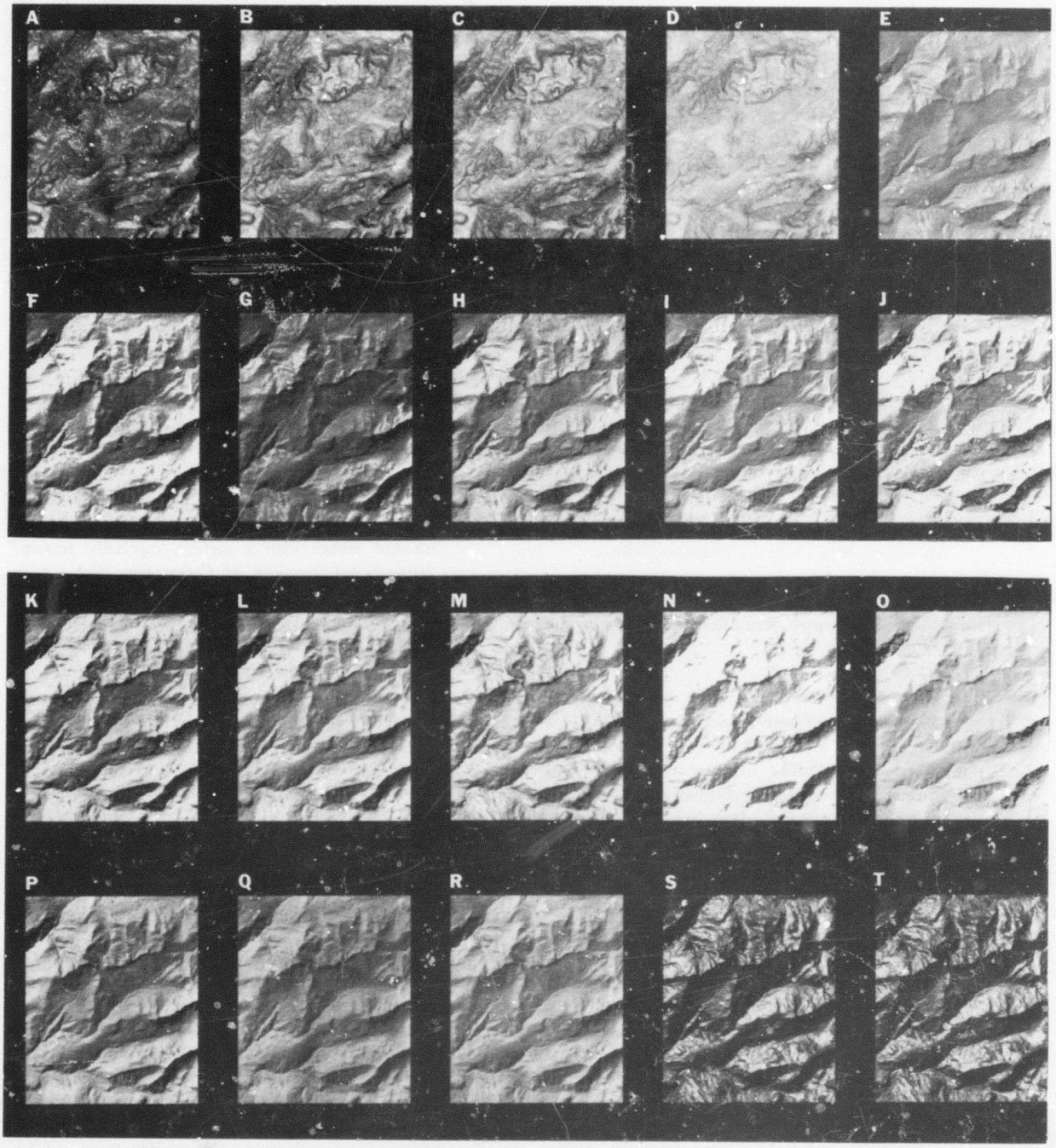


Figure 23: Relief shading produced using the reflectance maps shown in the previous figure. The order is by rows from top to bottom. Within each row the order is from left to right. The digital terrain model used has 200 columns of 240 rows.

References

[1] Imhof, E. (1965) *Kartographische Geländedarstellung*, W. de Gruyter & Co., Berlin, 425 pages.

[2] Lehmann, J. G. (1799) *Darstellung einer neuen Theorie der Bezeichnung der schiefen Flächen im Grundriss oder der Situationzeichnung der Berge*, Leipzig.

[3] Lehmann, J. G. (1816) *Die Lehre der Situations-Zeichnung oder Anweisung zum richtigen Erkennen und genauen Abbilden der Erd-Oberfläche in topographischen Charten und Situation-Planen*, Arnoldische Buch- und Kunsthandlung, Dresden.

[4] Chauvin, F. (1852) *Die Darstellung der Berge in Karten und Plänen, mit besondered Rücksicht auf ihre Anwendbarkeit im Felde*, Nauck'sche Buchhandlung, Berlin.

[5] Chauvin, F. (1854) *Das Bergzeichnen rationell entwickelt*, Nauck'sche Buchhandlung, Berlin.

[6] Vogel, C. (1893) "Die Terraindarstellung auf Landkarten mittels Schraffierung," *Petermanns Geogr. Mitt.*, Vol. 39, pp 148.

[7] Bach, H. (1853) *Die Theorie der Bergzeichnung in Verbindung mit Geognosie*, Stuttgart.

[8] Wiechel, H. (1878) "Theorie und Darstellung der Beleuchtung von nicht gesetzmässig gebildeten Flächen mit Rücksicht auf die Bergzeichnung", *Civilingenieur*, Vol. 24, pp 335—364.

[9] Burmester, L. (1875) *Theorie und Darstellung gesetzmässig gestalteter Flächen*, Leipzig.

[10] Tanaka, K. (1930) "A New Method of Topographical Hill Delineation," Memoirs of the College of Engineering, Kyushu Imperial University, Fukuoka, Japan, Vol. 5, No. 3, pp 121—143.

[11] Tanaka, K. (1932) "The orthographical relief method of representing hill features on a topographical map", *Geographical Journal*, Vol. 79, No. 3, (March) pp 213—219.

[12] Winterbotham, H. StJ. L. (1932) "Note on Professor Kitiro's Method of Orthographical Relief", *Geographical Journal*, Vol. 80, pp 518—520.

[13] Wilski, P. (1934) "Eine neue Japanische Darstellung der Höhen auf Landkarten," *Petermanns Mitteilungen*, Vol. 80, pg 359.

[14] Remiszewska, Jadwiga (1955) "Metoda cieć pochylych w kartograficznym obrazie urzeźbienia" [The Method of Inclined Cut in the Cartographic Presentation of the Form of the Land], *Polish Geographical Review*, Vol. 27, pp 125—134

[15] Robinson, A. H. & Thrower, N. J. W. (1957) "A new method for terrain representation," *Geographical Review*, Vol. 47, No. 4, (October) pp 507—520.

[16] Robinson, A. H. (1961) "The Cartographic Representation of the Statistical Surface," *International Yearbook of Cartography*, Vol. 1, pp 53—63.

[17] Thrower, N. J. W. (1963) "Extended uses of the method of orthogonal mapping of traces of parallel, inclined planes with a surface, especially terrain," *International Yearbook of Cartography*, Vol. 3, pp 26—28.

[18] King, C. M. (1966) *Techniques in Geomorphology*, St. Martin's Press, New York, pp 255—256.

[19] Peucker, T. K., Tichenor, M. & Rase, W.-D., (1972) "Die Automatisierung der Methode der schrägen Schnittflächen," *Kartographische Nachrichten*, Vol. 22, No. 4, pp 143—148.

[20] Oberlander, T. M. (1968) "A Critical Appraisal of the Inclined Contour Technique of Surface Representation", *Annals, Association of American Geographers*, Vol. 58, No. 4, pp 802—813.

[21] Robinson, A. H. & Sale, R. D. (1969) *Elements of Cartography*, 3rd ed., John Wiley, New York, pp 189—196.

[22] Robinson, A. H. & Thrower, N. J. W., (1969) "On surface representation using traces of parallel inclined planes," *Annals, Association of American Geographers*, Vol. 59, No. 3. pp 600—603.

[23] Oberlander, T. M. (1969) "Reply to Robinson-Thrower Commentary," *Annals, Association of American Geographers*, Vol. 59, No. 3, pp 603—605.

[24] Tanaka, K. (1939) "The Relief Contour Method of Representing Topography on Maps," *The Geographical Review of Japan*, Vol. 15, No. 9 & 10, pp 655—671, 784—796 (Japanese) pg 797 (English abstract).

[25] Tanaka, K. (1950) "The Relief Contour Method of Representing Topography on Maps," *Geographical Review*, Vol. 40, No. 3, pp 444—456.

[26] Tanaka, K. (1951) "The Relief Contour Method of Representing Topography on Maps," *Surveying and Mapping*, Vol. 11, pg 27.

[27] Köpcke, C. (1885) "Ueber Reliefs und Relief-Photogramme," *Civilingenieur*, Vol. 31.

[28] Pauliny, Y. (1895) "Memoire über eine neue Situations Pläne- und Landkartendarstellungs methode," *Streffen's Oesterreichischer Militärische Zeitschrift*, Vol. 36, pg 177.

[29] Raisz, E. J., (1931) "The Physiographic Method of Representing Scenery on Maps," *Geographical Review*, Vol. 21, No. 2, (April), pp 297—304.

[30] Raisz, Erwin (1938) *General Cartography*, New York and London.

[31] Pillewizer, W. (1957) "Geländedarstellung durch Reliefphotographie," *Kartographische Nachrichten*, Vol. 7, pg 141.

[32] Stoessel, O. C. (1959) "Photomechanische Reliefschummerung," *Nachrichten aus dem Karten- und Vermessungswesen*, Vol. 1, No. 10, pp 53—55.

[33] Noma, A. A., & Misulia, M. G. (1959) "Programming Topographic Maps for Automatic Terrain Model Construction," *Surveying and Mapping*, Vol. 19, No. 3, (Sept.) pg 335.

[34] Wilkerson, H. R. (1959) "Reliefschummerung durch Photographie von Geländemodellen, *Nachrichten aus dem Karten- und Vermessungswesen*, Vol. 1, No. 10, pp 60—62

[35] Friedemann, H. (1962) "Von neuen Erfindungen: Anordnung und Verfahren zur Herstellung von Schummerungen für kartographische Zwecke," *Kartographische Nachrichten*, Vol. 12, pg 150.

[36] Richarme, P. (1963) "L'estompage photographique," *Bulletin du comité francais de cartographie*, Vol. 17, pg 188.

[37] Richarme, P. (1963) "The Photographic Hill Shading of Maps", *Surveying and Mapping*, Vol. 23, No. 1, pp 47—59.

[38] Gilman, C. R. (1971) "Photomechanical Experiments in Automated Cartography," *Proc. ASM Fall Meeting 1971*, San Fransisco.

[39] Lyons, H. G. (1909) *The Representation of Reliefs on Maps*, Ministry of Finance, National Printing Department, Egypt.

[40] Raisz, E. J. (1956) "Landform Maps," *Petermanns Geogr. Mitt.*, Vol. 100, 1956, pp 171—172.

[41] Harris, L. J. (1959) *Hill-shading for relief-depiction in topographical maps*, London.

[42] Imhof, E. (1947) "Geländedarstellung in Karten grosser und mittlerer Massstäbe," *Vortrag Natf. Ges. Zürich*, (Jan).

[43] Carlberg, B. (1954) "Schweizer Manier und wirklichkeitsnahe Karte, Probleme der Farbgebung," *Kartographische Nachrichten*, Vol. 4, pp 8—14

[44] Pöhlmann, G. (1958) "Heutige Methoden und Verfahren der Geländedarstellung," *Kartographische Nachrichten*, Vol. 8, No. 3, pp 71—78.

[45] Mietzner, H. (1959) "Die Schummerung unter Annahme einer naturgemässen Beleuchtung," *Kartographische Nachrichten*, Vol. 9, No. 3, pp 73—79.

[46] Imhof, E. (1959) "Probleme der Kartographischen Geländedarstellung," *Nachrichten aus dem Karten- und Vermessungswesen*, Vol. 1, No. 10, pg 9—31.

[47] Keates, J. S. (1961) "Techniques of Relief Representation," *Surveying and Mapping*, Vol. 21, No. 4, (Dec.) pp 459—463.

[48] Hölzel, F. (1962) "Die Geländeschummerung in der Krise?", *Kartographische Nachrichten*, Vol. 12, No. 1, (Feb.) pp 17—21.

[49] Hölzel, F. (1963) "Generalization Problems in Hill Shading," *Nachrichten aus dem Karten- und Vermessungswesen*, Vol. 5, No. 5, pp 23—33.

[50] Imhof, E. (ed) (1963). *International yearbook of cartography*, George Philip and Son. London. 231 pages.

[51] Yoéli, P. (1959) "Relief Shading," *Surveying and Mapping*, Vol. 19, p 229.

[52] Yoéli, P. (1965) "Analytische Schattierung," *Kartographische Nachrichten.*, Vol. 14, No. 4, pp 142—148.

[53] Yoéli, P. (1965) "Analytical Hill Shading," *Surveying and Mapping*, Vol. 25, No. 4 (Dec.) pp 573—579.

[54] Yoéli, P. (1966) "Analytische Schattierung und Dichte," *Kartographische Nachrichten*, Vol. 16, No. 1, pp 17—23.

[55] Yoéli, P. (1966) "Analytical Hill Shading and Density, *Surveying and Mapping*, Vol. 26, No. 2 (June) pp 253—259.

[56] Yoéli, P. (1966) "Die Mechanisierung der Analytischen Schattierung," *Kartographische Nachrichten*, Vol. 16, No. 3, pp 103—107.

[57] Yoéli, P. (1967) "The Mechanisation of Analytical Hill Shading," *The Cartographic Journal*, Vol. 4, No. 2 (Dec.),

[58] Yoéli, P. (1967) "Die Richtung des Lichtes bei analytischer Schattierung," *Kartographische Nachrichten*, Vol. 17, No. 2, pp 37—44.

[59] Yoéli, P. (1971) "An Experimental Electronic System for Converting Contours into Hill-shaded Relief," *International Yearbook of Cartography*, Vol. 11, pp 111—114.

[60] Blachut, T. J., Marsik, Z., & Makow, D. (1969) "Relief shading process," Canada Patent No. 051-814.

[61] Marsik, Z. (1971) "Automatic Relief Shading," *Photogrammetrica*, Vol. 27, No. 2, pp 57—70.

[62] Peucker, T. K., (1972) "Computer Cartography," Association of American Geographers, Washington, D.C. . Commission on College Geography. *Resource Paper No. 17.*, pp 41—54.

[63] Peucker, T. K. & Cochrane, D. (1974) "Die Automation der Reliefdarstellung — Theorie und Praxis," *International Yearbook of Cartography*, Vol. 14, pp 128—139.

[64] Peucker, T. K., Tichenor, M. & Rase, W.-D. (1974) "The Computer Version of Three Relief Representations," In: Davis, J. C. and McCullagh, M. (eds.) *Display and Analysis of Spatial Data*, John Wiley, New York.

[65] Eckert, M. (1962) *Die Kartenwissenschaft: Forschung und Grundlagen zu einer Kartographie als Wissenschaft*, Vol. 1, Walter de Gruyt, Berlin.

[66] Carmichael, L. D. (1964) "Experiments in Relief Portrayal," *Cartographic Journal*, Vol. 1, pp 11—17.

[67] Eckert, M. (1965) *Die Kartenwissenschaft: Forschung und Grundlagen zu einer Kartographie als Wissenschaft*, Vol. 2, Walter de Gruyt, Berlin.

[68] Monmonier, M. S. (1965) "The Production of Shaded Maps on the Digital Computer," *The Professional Cartographer*, Vol. 17, No. 5, (Sept) pp 13—14.

[69] Koldayev, P. K. (1967) "Plastic Colour and Shadow Relief Representation," *Academy of Sciences of the USSR*, Council of Soviet Cartographers, Moscow.

[70] Sprunt, B. F. (1969) "Computer-generated halftone images from digital terrain models" MSc Dissertation, Dept. Mathematics, Univ. Southampton, 80p.

[71] Anda, B. (1974) "Automatic Hill-Shading using an Automatic Flatbed Drafting Machine with a Standard Photohead," *ITC Journal*, Enschede, No. 2, pp 212—216.

[72] Batson, R. M., Edwards, E. & Eliason, E. M. (1975) "Computer Generated Relief Images," *Journal of Research, U. S. Geological Survey*, Vol. 3, No. 4, (July-August), pp 401—408.

[73] Brassel, K. (1971) "Darstellungsversuche mit dem Datengesteureten Schnelldrucker," *Kartographische Nachrichten*, Vol. 21, No. 5, (Oct.) pp 182—188.

[74] Brassel, K. (1973) "Modelle und Versuche zur automatischen Schräglichtschattierung," Ph. D. dissertation, Geography Department, University of Zurich, Klosters, Switzerland, 112p.

[75] Brassel, K. (1973) "Ein-und mehrfarbige Printerdarstellungen," *Kartographische Nachrichten*, No. 5, pp 177—183.

[76] Brassel, K. (1974) "Ein Modell zur Automatischen Schräglichtschattierung," *International Yearbook of Cartography*, pp 66—77.

[77] Brassel K. (1974) "A Model for Automated Hill Shading," *The American Cartographer*, Vol. 1, No. 1, (April) pp 15—27.

[78] Blascke, W. (1967) "Le Modele Digital M.I.T.," *Societe Francaise de Photogrammetrie*, Bulletin 27, (July), pp 37—40.

[79] Bloehm, B. W. (1967) "Tabular representations of multi-variate functions — with applications to topographic modelling," Assoc. Comp. Machinery, 22nd Nat. Conf. Proc., pp 403—415.

[80] Aumen, W. (1970) "A New Map Form: Numbers," *International Yearbook of Cartography*, Vo. 10, pp 80—84

[81] Grist, M. W. (1972) "Digital Ground Models: An Account of Recent Research," *Photogrammetric Record*, Vol. 70, No. 4, (Oct.) pp 424—441.

[82] Silar, F. (1972) "Das digitale Geländemodell — Theorie und Praxis," *Vermessungstechnik*, Vol. 20, No. 9, pp 327—329.

[83] Torlegard, K. (1972) "Digital Terrain Models — General Survey and Swedish Experiences," *Bildmessung und Luftbildwesen*, Vol. 40, No. 1, pp 21—30.

[84] American Society of Photogrammetry, (1978) *Proc. of the Digital Terrain Models (DTM) Symposium*, St. Louis Missouri (May 9-11).

[85] Wild Heerburg & Raytheon, (1968), "B8 Stereomat Automated Plotter," company sales literature.

[86] Bertram, S. (1969) "The UNIMACE and the Automatic Photomapper," *Photogrammetric Engineering*, Vol. 35, pp 569—576.

[87] Babcock, H. C. (1970) "Evaluation of a Stereocompilation Digitizer," *Congress on Surveying and Mapping*, 30th Annual Meeting, pp 338—347

[88] Seymour, R. H. & Whiteside, A. E. (1972) "A new Computer-Assisted Stereocomparator," *Bendix Technical Journal*, (Spring), pp 1—5.

[89] Crawley, B. G. (1974) "Gestalt Contours," *The Canadian Surveys*, Vol. 28, No. 3, (Sept.), pp 237—246.

[90] Bendix Research Laboartories, (1976) "AS-11B-X Automatic Stereo Mapper," RADC-TR-76-100, Rome Air Development Center, Griffiss Air Force Base, New York, (April).

[91] Panton, D. J. (1976) "Digital Stereo Mapping," *Countermeasures*, (May), pg 12.

[92] Loscher, W., (1967) "Some aspects of orthophoto technology," *Photogrammetric Record*, Vol. 6, No. 30, pp 419—432.

[93] Blachut, T. J. (1968) "Further extension of the orthophoto technique," *Canadian Surveyor*, Vol. 22, No. 1, pp 206—220.

[94] Blachut, T. J. & Van Wijk, M. C. (1970) "3-D Information from Orthophotos," *Photogrammetric Engineering*, (April), pp 365—376.

[95] Hughes, T. A., Shope, A. R. & Baxter, F. S. (1971) "USGS Automatic Orthophoto System," *Photogrammetric Engineering*, Vol. 37, pp 1055—1062.

[96] Beyer, A. (1972) "Zur Erfassung flächen Geländes durch willkürlich verteilte Höhenpunkte," *Vermessungstechnik*, Vol. 20, No. 6, pp 204—207

[97] Peucker, T. K. & Chrisman, N. (1975) "Cartographic Data Structures," *The American Cartographer*, Vol. 2, No. 1, (April), pp 55—69.

[98] Keppel, E. (1975) "Approximating complex surfaces by triangulation of contour lines," *IBM Journal of Research and Development*, (January).

[99] Peucker, T. K., Fowler, R. J., Little, J. J. & Mark, D. M. (1976) "Digital Representation of Three-Dimensional Surfaces by Triangulated Irregular Networks (TIN)," Technical Report No. 10 (Revised), Dept. of Geography, Simon Fraser University, Barnaby, B. C., Canada.

[100] Goosen, D. A. (1962) "Blockdiagrams," *ITC Information*, No. 3, pg 20.

[101] Jenks, G. F. & Brown, D. A. (1966) "Three-Dimensional Map Construction," *Science*, Vol. 154, No. 3750, pp 857—864.

[102] Douglas, D. (1971) "VIEWBLOK: A computer program for constructing perspective view block diagrams," *Revué de Geographie de Montreal*, Vol. 26, p. 102—104.

[103] Wright, T. J. (1973) "A Two-Space Solution to the Hidden Line Problem for Plotting Functions of Two Variables," *IEEE Trans. on Computers*, Vol. C-22, No. 1, (January) pp 28—33.

[104] Warnock, J. E. (1969) "A hidden-surface algorithm for computer generated half-tone pictures," TR 4-15, Dept. of Computer Science, University of Utah, Salt Lake City, Utah.

[105] Watkins, G. S. (1970) "A real-time visible surface algorithm," Report UTEC-CSC-70-101, (June), Dept. of Computer Science, University of Utah, Salt Lake City, Utah.

[106] Bouknight, W. J. (1970) "A Procedure for Generation of Three-Dimensional Half-toned Computer Graphics Presentations," *Commun. of the A.C.M.*, Vol. 13, No. 9, (Sept) pp 527—536.

[107] Goldstein, R. A. & Nagel, R. (1971) "3-D Visual Simulation," *Simulation*, Vol. 16, pp 25—31.

[108] Gouraud, H. (1971) "Computer display of curved surfaces," *IEEE Trans. on Computers*, Vol. C-20, (June) pg 623—629.

[109] Newell, M. E., Newell, R. G. & Sancha, T. L. (1973) "A New Approach to the Shaded Picture Problem," *Proc. of the ACM National Conference*, Boston, Mass, Vol. 1, pp 443—450.

[110] Staudhammer, J. & D. J. Odgen (1975) "Computer Graphics for Half-tone three-dimensional Object Images," *Compt. & Graphics*, Vol. 1, No. 1, pp 109—114.

[111] Catmull, E. A. (1975) "Computer display of curved surfaces," *Proc. IEEE Conf. Computer Graphics, Pattern Recognition and Data Structures*, Los Angeles, (May) pp 11—17, (IEEE Cat. No. 75CH0981-1C).

[112] Bui-Tuong, Phong (1975) "Illumination for computer-generated images," *Commun. of the A.C.M.*, Vol. 18, No. 6, (June) pp 311—317.

[113] Blinn, J. F. & Newell, M. E. (1976) "Texture and reflection in computer generated images," *Commun. of the A.C.M.*, Vol. 19, No. 10, (Oct) pp 542—547.

[114] Blinn, J. F. (1977) "Models of light reflection for computer synthesized pictures," *SIGGRAPH '77, Proc. 4th Conference on Computer Graphics and Interactive Techniques, A.C.M.*, pp 192—198.

[115] Blinn, J. F. (1978) "A Scan Line Algorithm for Displaying Parametrically Defined Surfaces," *SIGGRAPH '78, Proc. 5th Conference on Computer Graphics and Interactive Techniques, A.C.M.*.

[116] Lambert, J. H. (1760) *Photometria sive de mensura et gratibus luminus colorum et umbrae*, Eberhard Klett, Augsburg.

[117] l'Abbe de Lacaille (1760) *Traite d'optique sur la gradation de la lumière*, (Ouvrage posthume de M. Bouguer), Paris.

[118] Lommel, (1880) "Ueber Flourescence," *Annalen der Physik*, Leipzig, Vol. 10, pg. 449.

[119] Seeliger, (1888) "Die Photometrie von diffus reflektierenden Flächen," *S. B. Bayer. Akad. Wiss.*, Vol. 18, pg. 20.

[120] Markov, A. (1924) "Les particularités dans le réflexion de la lumière par la surface de la lune," *Astronomische Nachrichten*, Vol. 221, pp 65—78.

[121] Schönberg, E. (1925) "Untersuchungen zur Theorie der Beleuchtung des Mondes auf Grund photometrischer Messungen," *Acta Soc. Sci. Fennicae*, Vol. 50, pp 1—70.

[122] Fesenkov, V. G. (1929) "Photometric Investigations of the Lunar Surface," *Astronomicheskii Zhurnal*, Vol. 5, pp 219—234.

[123] Minnaert, M. (1941) "The Reciprocity Principle in Lunar Photometry," *Astrophysical Journal*, Vol. 93, pp 403—410.

[124] Fedoretz, V. A. (1952) "Photographic Photometry of the Lunar Surface," *Publ. Kharkov Obs.*, Vol. 2, pp 49—172.

[125] Minnaert, M. (1961) "Photometry of the Moon," Chapter 6 in: Kuiper, G. P. & Middlehurst, B. M. (eds.) *Planets and Satellites*, Univ. Chicago Press, Vol. 3, pp 213—248.

[126] Fesenkov, V. (1962) "Photometry of the Moon," in: Kopal, Z. (ed.) *Physics and Astronomy of the Moon*, Academic Press, New York, pp 99—130.

[127] Hapke, B. W. (1963) "A Theoretical Photometric Function for the Lunar Surface," *Journal of Geophysical Research*, Vol. 68, No. 15, pp 4571—4586.

[128] Hapke, B. & Van Horn, H. (1963) "Photometric Studies of Complex Surfaces, with Applications to the Moon," *Journal of Geographical Research*, Vol. 68, No. 15, pp 4545—4570.

[129] Hapke, B. (1966) "An Improved Theoretical Lunar Photometric Function," *The Astronomical Journal*, Vol. 34, No. 3, (March) pp 560—570.

[130] Middleton, W. E. & Mungall, A. G. (1952) "The Luminous Directional Reflectance of Snow," *Journal of the Optical Society of America*, Vol. 42, No. 3, pp 572—579.

[131] Planck, M. (1959) *The Theory of Heat Radiation*, Dover, New York.

[132] Beckmann, P. & Spizzichnio, A. (1963) *The Scattering of Electromagnetic Waves from Rough Surfaces*, Pergamon Press, New York.

[133] Torrance, K. E., Sparrow, E. M. & Birkebak, R. C. (1966), "Polarization, directional distribution, and off-specular peak phenomena in light reflected from roughened surfaces," *Journal of the Optical Society of America*, Vol. 56, No. 7, (July) pp 916—925.

[134] Torrance, K. E. & Sparrow, E. M. (1967) "Theory for off-specular reflection from roughened surfaces," *Journal of the Optical Society of America*, Vol. 57, No. 9, (Sept.) pp 1105—1114.

[135] Trowbridge, T. S. & Reitz, K. P. (1975) "Average irregularity representation of a rough surface for ray reflection," *Journal of the Optical Society of America*, Vol. 65, No. 5, (May) pp 531—536. Photogrammetric Research, Ottowa, (June), pg 147.

[136] Horn, B. K. P. (1975) "Determining Shape from Shading," Chapter 4, in Winston, P. H. (ed): *The Psychology of Computer Vision*, McGraw-Hill.

[137] Horn, B. K. P. (1977) "Understanding Image Intensities," *Artificial Intelligence*, Vol. 8, No. 11, pp 201—231.

[138] Horn, B. K. P. & Bachman, B. L. (1978) "Using Synthetic Images to Register Real Images with Surface Models," *Commun. of the A.C.M.*, Vol. 21, No. 11, (Nov.) pp 914—924.

[139] Horn, B. K. P. & Sjoberg, R. W. (1979) "Calculating the Reflectance Map," *Applied Optics*, (to appear in June).

[140] Woodham, R. J. (1978) "Photometric Stereo: A reflectance map technique for determining surface orientation from image intensity," *Image Understanding Systems and Industrial Applications*, Proc. S.P.I.E., Vol. 155, (August),

[141] Nicodemus, F. E., Richmond, J. C. & Hsia, J. J., Ginsberg, I. W., Limperis, T. (1977) "Geometrical Considerations and Nomenclature for Reflectance," *NBS Monograph 160*, National Bureau of Standards, U. S. Department of Commerce, Washington, D. C., (October).

[142] Nicodemus, F. E. (ed.) (1976, 1977, 1978) "Self-Study Manual on Optical Radiation Measurements," *NBS Technical Notes 910-1, 910-2 & 910-3*, National Bureau of Standards, U. S. Department of Commerce, Washington, D. C.

[143] McCartney, E. J. (1976) *Optics of the Atmosphere: Scattering by Molecules and Particles*, John Wiley, New York.

[144] Conte, S. D. & de Boor, C. (1972) *Elementary Numerical Analysis*, McGraw-Hill.

[145] Hamming, R. W. (1962) *Numerical Methods for Scientists and Engineers*, McGraw-Hill.

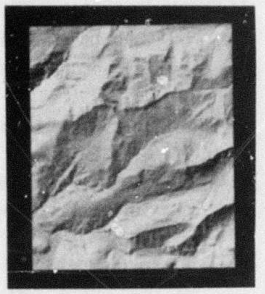
[146] Richtmeyer, R. D. & Morton, K. W. (1967) *Difference Methods for Initial-Value Problems*, John Wiley. pp 136—143.

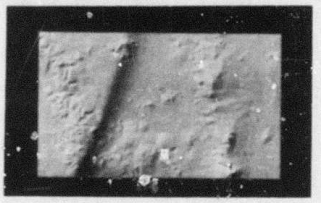
[147] Bridge, P. M. & Inge, J. L. (1972) "Shaded Relief of Mars," Atlas of Mars, MH 25 M IR, JPL Contract WO-8122, USGS, Department of the Interior.

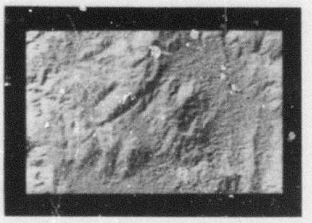
[148] Bantel, W. (1973) "Der Reproduktionsweg vom einfarbigen Relieforiginal zur mehrfarbigen Reliefkarte," *International Yearbook of Cartography*, Vol. 13, pp 134—136.

[149] DeLucis, A. (1971) "The effect of shaded relief terrain representation on map information accessibility," *American Congress on Surveying and Mapping*, 31st Annual Meeting, Washington, D.C., pp 641—657.

[150] Neumann, J. (1973) "Begriffsgeschichte und Definition des Begriffes 'Kartographische Generalisierung'," *International Yearbook of Cartography*, Vol 13, pp 59—67.







SESSION III

TECHNICAL PAPERS

STRIP TREES:
A HIERARCHICAL REPRESENTATION FOR MAP FEATURES

Dana H. Ballard

Computer Science Department
University of Rochester
Rochester, New York 14627

ABSTRACT

There is increasing interest in map features such as points, lines and regions both as a pictoral data base for resource management and as an aid to identifying objects in aerial images. Owing to the very large amount of data involved, and the need to perform operations on this data efficiently, the representation of such features is a crucial issue. We describe a hierarchical representation of map features that consists of binary trees with a special datum at each node. This datum is called a strip and the tree that contains such data is called a strip tree. Lower levels in the tree corresponds to finer resolution representations of the map feature. The strip tree structure is a direct consequence of using the method for digitizing lines given by [Duda & Hart, 1973; Turner, 1974; Douglas-Peucker, 1973] and retaining all intermediate steps. This representation has several desirable properties. For features which are well-behaved, calculations such as point-membership and intersection can be resolved in $\emptyset(\log n)$ where n is the number of feature points. The map features can be efficiently encoded and displayed at various resolutions. The representation is closed under intersection and union and these operations can be carried out at different resolutions. All these properties depend on the hierarchical tree structure which allows primitive operations to be performed at the lowest possible resolution with great computational savings. The strip tree representation also can allow parts of the map feature to be accessed sequentially. This feature is usually desired when the map feature is used in analyzing images.

The price paid for the improved performance is an increased storage cost. This is approximately 4n, where n is the storage needed to represent the xy coordinates.

1. Introduction

We present a general representation for polylines (connected line segments) and areas (closed polylines). Although this representation may have wide applications, its principal motivation arose from the problem of representing geographical data bases of map features.

A map has several interesting kinds of features such as contour lines, lakes, rivers, roads, etc. These can be roughly divided into four feature classes for representation in the computer [Sloan, 1978]:

feature	examples in map domain
points	towns (large scale maps)
	bridges (small scale maps)
lines	roads, coastlines
strips	wide roads, rivers
regions	lakes, counties

Our main interest is in representing lines and regions. A point is such a simple datum that it can be easily treated as a primitive in any representation. Collections of points from a single class can be efficiently represented as k-d trees [Bently, 1975; Barrow et.al., 1977] and so points are not the focus of our interest, although they do interact with our representation. A strip feature is essentially a line where a locally varying thickness is important, examples of which are rivers and roads. As we shall see, our representation for lines will also encompass this type of feature.

We regard collections of these map features as a data base that might be used to perform the following tasks:

.Find where a road intersects a river

.Display a subset of map features that appear in a given map sector

.Find out if a given point is in a region

.Search an aerial image near the edge of a dock for ships.

A very important aspect of all these tasks is that we may be satisfied if they are performed at resolution lower than the ultimate resolution represented.

Our representation for lines and regions consists of a binary tree structure where, in general, lower levels in the tree correspond to finer resolutions. The tree structure is a direct consequence of using the method for digitizing lines given by [Duda and Hart, 1973; Turner, 1974] and retaining all intermediate steps in the digitization process. As an example of the representation, Figure 1 shows some roads represented at various levels (resolutions) in the tree structure.

The idea of representing a line by sets of strips was recognized by [Peucker, 1976]. In particular he was able to find line intersection and point in polygon algorithms. However, the tree structure is a vast improvement over the set organization: the algorithms are more efficient, line-area interesection and area-area intersection and union can now be dealth with, and the tree structures are closed under these operations.

Figure 1. Map features displayed at various resolutions using the hierarchical structure.

2. The Strip Tree

2.1 Notation

We define a strip segment $\underline{L}$(delta) as the vector $\underline{L}$ and the scalar delta as shown by Figure 2. The vector $\underline{L}$ starts at (XBeg,YBeg) and ends at (XEnd,YEnd). We use S to denote the set of points inscribed by the rectangle defined by $\underline{L}$(delta). Also we denote the boundaries of the rectangle by the line segments 1+, 1-, e+, e- as shown.

Figure 2. Definition of a Strip Segment.

A polyline is an ordered list of discrete points y0,...,yn subsets of which may be colinear. For the moment we require these points to be considered as connected; later we will relax this condition. We say a polyline is represented at resolution delta* if there exists an ordered sequence of m strip segments

$$\underline{L}_k(\text{delta}),\ k=0,\ \ldots,\ m-1$$

such that

$$\text{delta} < \text{delta}^*\ k=0,\ \ldots,\ m$$

$$yi \in \bigcup_{k=0}^{m} \underline{L}_k\ i=1,\ \ldots,n$$

If within a strip segment there is a point y that is a member of e+, another that is a member of e-, and there is a point y that is a member of 1+ and another that is a member of 1-, then the strip segment is said to be compact. The compactness property is very important for some of the algorithms which follow. Figure 1 shows some examples for different deltas.

2.2. Digitization

Suppose we have a polyline P. such as shown by Figure 3a. For any resolution delta we can approximate this line with strip segments as follows [Duda & Hart, 1973; Turner, 1974]:

> Consider the polyline P defined by {y0,...,yn}. For each point y P find the perpendicular distance d(y) from y to P. Denote the subset of y P such that $\underline{y}.\underline{L}>0$ as P+. P-= P-P+. Now find d+ = max d(y) and d- = max d(y). If (d+) + (d-) <delta* then the polyline is compactly represented at resolution by the strip tree consisting of a single root strip 1((d+)+(d-)). If not then the desired strip tree is obtained by recursively applying the algorithm to the Ps y0,...,y+ and (y+)+1,...,yn and making the results the left son and right son respectively of the

strip tree. In the case of ties for the maximum distance d, we will arbitrarily pick the point nearest the mid point (in arc length).

For the purposes of the union and intersection algorithms to follow it is helpful to think of the strip trees as completely expanded down to individual points, even though these points may be colinear. Figure 3 shows an example of two levels of recursion of this algorithm.

Figure 3. Steps in the Digitization Process.

To see formally that the convergence is guaranteed, note that a P of k points can always be approximated by a single strip segment L(k) with length k assuming eight-connectedness. Thus for any delta there must be a strip tree with leaves consisting of no more than 2n/delta strip segments which approximate P. Since the digitization algorithm splits each P into two parts such that each part has finite length, the process must ultimately consider sets of P of delta points or less.

2.3 Strip Tree definitions

The binary tree resulting from the digitization process is called a strip tree, where the datum at each node is a strip, $\underline{L}$. The nodes of the tree are initially ordered on arc length. (Later we will see that when intersection occurs in two areas which are represented in strip trees, this property is sometimes not preserved).

In the ensuing algorithms we will use the following definitions:

T = symbol for a Strip Tree obtained by the digitization process.

S(T) = the points associated with the strip at the root node of T; i.e. $\{x | x \in S(T)\}$

Area(T) = the area associated with the strip at the root node of T. We measure area in pixels so that a strip $\underline{L}(0)$ still has finite area. The most primitive strip, a single point has unit area.

LSon(T) = the left son of the node T

RSon(T) = the right son of the node T

A node of the strip tree is completely defined by the seven-tuple (LSon, RSon, Area, XBeg, XEnd, YBeg, YEnd). The measure Area(T) is better for some of the algorithms to follow. Area and delta are related by delta $\approx$ Area/$||\underline{L}||$.

2.4. Why Binary Trees?

The polylines can also be represented as a tree with nodes of more than two siblings. In fact, nodes could have different numbers of siblings which would still be ordered. Figure 4 shows an example of the alternate encoding scheme. In certain cases this may be a more concise representation for the polyline and for all the algorithms tht follow we can extend the operations from two sons to multiple sons. However, this change does not alter the complexity of the operations that we would like to perform and can be more inefficient than the binary tree representation.

Figure 4. A portion of an encoding using m-ary trees.

3. Operations on Polylines

Computational complexity of the various operations is difficult to characterize, as it depends on the particular geometry of polylines. If the polylines are "well-behaved", that is they are relatively smooth and do not self-intersect for more than a few points, then the algorithms are very efficient. What this means for a particular operation in terms of the strip tree is that if the number of strips that must be examined at any level is constant, then the complexity of the operation is $\emptyset(\log n)$.

3.1. Testing the Proximity of a Point

If we would like to find out if a point is near a polyline, this may be discovered early using the strip tree. We can make this more precise by exploiting the following property:

Property P1:

A. If a point z is inside a compact strip 1(delta) then it can be at most $2\sqrt{2}$ units away from the P.

B. If a point z is outside a compact strip 1(delta) then the distance of the point from the P is bounded by

$$0 \leq z \leq d_S(z, 1(\text{delta})) + 2*\text{delta}$$

It is interesting to study these bounds as the depth in the resolution tree increases. Although the convergence is not monotonic, the bounds do converge to the actual set-theoretic distance $d_s(z,P)$. Now suppose we want to answer the question: is $d_s(z,P)<d_0$? If this can be answered affirmatively we will find this out at the point where any upper bound is less than d . If the answer is no, then this will be discovered when the tree has been explored to the point where all minimum bounds are greater than d_0. Similar arguments can be made for the qualitative level-of-effort required to answer: is $d_s(z,P)>d_0$? From this discussion we can see that the search will be inefficient only if $d_0 \approx d_s(z,S(T))$ and a large number of the strips are nearly d_0 from z. Figure 5a shows this case together with a more representative example.

Figure 5. Two of many Possible Geometrics When Testing the Distance of a Point from an P.

To summarize this discussion, we provide the algorithms to test for $d_s(z,P) < d_0$ and $d_s(z,P) > d_0$. These algorithms use the notion of the distance of a point to a set which is defined as follows. For any strip S, if a point is outside S i.e. $x \notin S$ then its distance to S is characterized by the set theoretic distance $ds(z,S) = \min_{x \in S} d(x,z)$ where d is the euclidean distance between the points x and z. For clarity, the algorithms are presented as procedures in a pseudo-Algol language. Rigor has been sacrificed mainly in the specification of data types, but these should be obvious from the earlier definitions.

Algorithm A1: Is a point within d0 of a polyline?

```
boolean procedure Within (z,d0,T)
   begin
   if d0≤ ds(z,S(T)) + 2.delta(T)
   then return (true);
   if z ∉ S(T) and d0>ds(z,S(T))
   then return (false);
   return (Within (z,d0,LSon(T))
   or Within (z,d0,RSon(T)));
   end;
```

Algorithm A2: Is a point further than d0 from a polyline?

```
boolean procedure Further
   (z,d0,T)
   begin
   if d0≤ ds(z,S(T)) + 2.delta(T)
   then return (false);
   if z ∉ S(T) and d0>ds(z,S(T))
   then return (true);
   return (Further (z,d0,LSon(T))
   and Further (z,d0,RSon(T)));
   end;
```

3.2 Displaying a Polyline at Different Resolutions

As previously demonstrated in Section 2, a polyline may be represented as a set of strip segments such that each strip segment L has a resolution delta less than some fixed delta0. The algorithm to display such a representation using the strip tree is as follows. This algorithm uses a device-dependent subroutine DisplayRectangle which paints the rectangle on the particular display device.

Algorithm A3: Display a polyline at Resolution delta0

```
procedure PolyDisplay (T,delta0)
   begin
   if delta(T) ≤ delta0 then
   DisplayRectangle
   (L(T),delta(T))
      else (PolyDisplay
   (LSon(T),delta0) and
   PolyDisplay (RSon(T),delta0));
   end;
```

3.3 Intersecting Two Polylines

One of the important features of the representation is the ability to compute intersections between polylines. Strip trees provide the facility to not only compute intersection points, but, in the case where lower resolution is satisfactory, to compute small areas containing the intersection points at great computational savings. In order to develop the intersection methodology, we need the following definitions:

A. Two strip segments (L1 derived from P1) and (L2 derived from P2) do not intersect iff $L1 \cap L2 = \emptyset$

B. Two strip segments L1, L2 have a clear intersection iff l1+ and l1- intersect l2+ and l2-.

C. Two strip segments L1 and L2 have a possible intersection if condition B is not satisfied yet $L1 \cap L2 \neq \emptyset$.

These cases are illustrated by Figure 6. A fairly obvious but very important lemma is:

Clear Intersection Lemma. [Peucker, 1976] If two strip segments have a clear intersection and the strips are both compact, then the corresponding Ps must also intersect.

To see this for condition B, consult Figure 6b. P1 divides the region R into two parts and P2 must cross from one to the other. The only way the P2 can do this is by intersecting P1.

Figure 6: Different Ways Strips can Intersect

The algorithms to check for intersections between two polylines are recursive, and assume the existence of an integer procedure StripIntersection which will return the type of intersection and, in the case of a clear type, will return a parallelogram Q containing the intersection points.

Algorithm A4: Finding out whether two polylines intersect

Comment. If the two root strip segments do not intersect then the Ps do not intersect. If the root segments have a clear intersection then the Ps intersect. Since the task is to just determine whether or not an intersection exists, we are done the moment we find a clear intersection.

```
boolean procedure Intersection
   (T1,T2, Primitive Flag)
   comment Primitive Flag allows
     the use of a single
      strip as the first argument
   begin
   Case StripIntersection
   (S(T1),S(T2),Q) into
      [Null] return (false),
      [Possible] if
   (Area(T1)>Area(T2)) or
   (Primitive Flag) then
         return
   ((Intersection(LSon(T1),T2) or
   (Intersection(RSon(T1),T2));
         else return
      (Intersection(T1,LSon(T2))
   or Intersection(T1,RSon(T2)));
      [Clear] return(true);
   end;
```

This procedure is easily modified to return a set of parallelograms comprising intersection points. Further easy modifications can be made to constrain these parallelograms to be of a certain size related to the delta(T1) and delta(T2); i.e., they can be made to be as small as we want.

Note, however, that smaller resolutions may be much more computationally expensive, as shown in the following example (Figure 7) where intersection at the coarsest resolution is simple, but multiple intersections occur at lower levels.

Figure 7: An intersection may be simple at one level and complicated at lower levels.

If the two Ps are not convoluted about each other the intersection will be computed in $\emptyset(m\log(n))$ steps where m is the number of intersection points. If the Ps do not intersect but have a closest distance d =ds(P1,P2) then this will be discovered at a level in the tree no deeper than a point where

$$d/2 > \text{delta1} + \text{delta2}$$

The worst case performance is intolerable as the algorithm's computation will grow exponentially as long as all the strip segments in one tree intersect all the strip segments in the other. In fact, the computation can be shown to be $\emptyset(2^K)$ where K is the sum of the depths in each tree where the comparisons are taking place! If this situation were encountered in a practical application, one way of handling it would be to report the possible intersection regions at the point where the limit of some bound on allotted resources was exceeded.

3.4 The Union of Two Polylines

The union of two strip trees can be accomplished by defining a strip that covers both of the two root strips.

Algorithm A5: P-P Union.
For two Ps defined by $\{y_0' \ldots y_n'\}$, $\{y_0'' \ldots y_m''\}$ treat these as two subsets and concatenate the subsets. That is, the resultant ordering is such that we have $y_0 = y_0'$, $y_{m+1} = y_m''$.

Now define a strip segment that covers $\{y_0, \ldots y_{m_{min}}\}$ such that c=0 and delta $=d^*$. By construction, this satisfies all the properties of a strip segment. Make this the root node of a new P-tree. The two subtrees are the two Ps of the union.

This construction is shown in Figure 8. The variable c is defined below.

Figure 8: Construction for Union of Strip Trees Representing Two Polylines

Of course this construction introduces a problem in that the new strip is no longer compact and therefore the Clear Intersection Lemma no longer holds. To overcome this problem we must add one bit of information to each node to mark whether the underlying polyline is compact. Since later algorithms may result in underlying polylines that are disconnected, we include this in the following definition of C:

C(T) = 1 P represented by S(T) is known to be compact and connected

· 0 otherwise

With this strategy we can preserve the eloquence of the previous algorithms in the following manner: When bit C(T) is not one we apply the recursion regardless of the intersection type. In algorithm A4 this means that clear intersections are reported as possible if the bit C(T) is set.

This technique can also be used as a digitization method for m non-connected segments

$\{(y_0^1, \ldots, y_{k_1}^1), (y_0^2, \ldots, y_{k_2}^2), \ldots, (y_0^m, \ldots, y_{k_m}^m)\}$

These segments are given an ordering as shown. The previous digitization algorithm is applied to this set of points, and the perpendicular distance d* is computed from the set of disconnected ys and used to define the delta of the root strip as before. However now the set is divided into two subsets of connected segments (rather than using y*) and the digitization algorithm is applied recursively to the subsets. Once this process produces connected subsets, the earlier digitization scheme is applied.

4. Areas Represented by Strip Trees

We take the boundary of an area to be a closed polyline. Interestingly enough, the digitization method described in Section 2 works for closed polylines and, incidentally, also for self-intersecting polylines. Furthermore, if an area is not simply connected it can still be represented as a strip tree, which at some level has connected primitives. The method for doing so was described in the previous section. If a region has holes it can be represented by a single boundary curve using a construction (Figure 9).

Figure 9: A Region with a Hole

If the holes are important, they themselves should be independently represented as strip trees.

The most remarkable fact is that by representing an area in this way many useful operations such as intersection between a polyline and an area, determining whether a point is inside an area, and intersecting two areas are carried out very efficiently.

4.1 Determining Whether a Point is Inside an Area

The strip tree representation of an area by its boundary allows the determination of whether a point is inside the area in a straightforward manner. If any semi-infinite line terminating at the point intersects the boundary of the area an odd number of times, the point is inside. This result appears in [Minsky and Papert, 1969]. This result is computationally simplified for strip trees in the following manner:

Point Membership Property

To decide whether a point z is member of an area represented by a strip tree, we need only compute the number of clear intersections of the strip tree with any semi-infinite strip L which has delta = 0 and emanates from z. If this number is odd then the point is inside the area.

An extension to the clear intersection lemma which makes this property hold is that the underlying curves may intersect more than once but must intersect an odd number of times. The following algorithm is used to determine whether a point is inside an area:

```
Algorithm A6:  Point Membership
      boolean procedure Inside(z,T)
         begin
         CreateStrip(S0,z)
         comment CreateStrip creates a
         strip for the half line.
         if
         NoOfClearIntersections(S0,T)
         is odd then return (true)
            else return (false);
         end;
      integer procedure
         NoOfClearIntersections(S,T)
         begin
         CaseStripIntersection(S,S(T))
         into
            [Null] return (0);
            [Possible] return
         (NoOfClearIntersections(S,LSon(
         T))
                  +
         NoOfClearIntersections
         (S,RSon(T)));
            [Clear] return (1);
         end;
```

A potential difficulty exists with the procedure NoOfClearIntersections when the strip SO is tangent to the polyline. Since this problem will only occur at the lowest level of the tree, we can examine neighboring leaves of the tree to resolve it.

4.2 Intersecting a polyline with an Area

The strategy behind intersecting a strip tree representing a polyline with a strip tree representing an area is to create a new tree for the portion of the polyline which overlaps the area. This can be done by trimming the original polyline strip tree. This is done efficiently by taking advantage of an obvious property of the intersection process:

Pruning Property: Consider two strips Sp e Tp and Sa e Ta. If the Sp∩Ta is null, then (a) if any point on Sp is inside Ta the entire tree whose root strip is Sp is inside or on Ta and (b) if any point on Sp is outside of Ta then the entire tree whose root strip is Sp is outside of Ta.

This leads to the recursive procedure A7 for polyline-area intersection using trees. Note that since strip nodes under a clear or possible strip intersection may be pruned, the bit c for the latter strip is set to 0 to denote that it no longer has the compactness property. Of course as repeated intersections are carried out with different areas more and more upper-level strips may have their bits set to 0; nevertheless, the intersected polyline is accurately represented at the leaves of the strip tree.

Note that if the polyline strip is "fatter," i.e., Area(T1) > Area(T2), we can copy the node and resolve the intersection at lower levels, whereas in the converse case we have to sequentially prune the tree by first intersecting the polyline strip with the left area strip and then intersecting the resultant pruned tree with the right area strip.

```
Algorithm      A7:        Polyline-Area
Intersection
     reference procedure
          PolyAreaInt(T1,T2)
       begin
       A:=T2
       comment A is a global used by
          PAInt;
       return(PAInt(T1,T2);
       end;


     reference procedure PAInt(T1,T2)
       begin
       Case StripInt(T1,T2) into
          [Null or Primitive]
             if Intersection (T1,A,
       TRUE) = null then
                   if Inside(T1,A) then
       return (T1)
                      else return (null);
             else return (T1);
```

```
[Clear or Possible] if
Area(T1)>Area(T2) then
   begin
   C(NT):=0
   comment non-compact strip
   XBeg(NT):= XBeg(T1);
   YBeg(NT):= YBeg(T1);
   XEnd(NT):= XEnd(T1);
   YEnd(NT):= YEnd(T1);
   Area(NT):= Area(T1);
   LSon(NT):= PAInt
(LSon(T1),T2);
   RSon(NT):= PAInt
(RSon(T1),T2);
   return(NT);
   end
else comment Area(T1)≤
Area(T2)
   Return
(PAInt(PAInt(T1,LSon(T2)),RSOn(
T2)));
end;
```

4.3 Intersecting Two Areas

The problem of intersecting two areas can be efficiently carried out using their strip tree representations. The method is to decompose the problem into two polyline area intersection problems (refer to Figure 10).

Figure 10: Decomposition of Area-Area Intersections

If we treat the boundary of A1 as representing a polyline instead of representing an area and intersect its strip tree with the strip tree representing A2 the lowest level result is shown by the thick lines in Figure 10a. If we reverse the roles of the two strip trees the result is given by the thick lines in Figure 10b. The union of these two strip trees (see Section 3.4) is the answer we want! Thus we can write the area-area intersection procedure in terms of strips as follows:

Algorithm A8: Area-Area Intersection

```
reference procedure AreaAreaInt (T1,T2)

        begin
        return (Union (PolyAreaInt
        (T1,T2)),(PolyAreaInt
        (T2,T1)));
        end;
```

where Union is a procedure that accomplishes the construction described in Section 3.4.

Note that in the case of areas that intersect in a way that fragments their boundaries, the order of the segments will not be preserved by the intersection procedure. (Until this point we were guaranteed that strips in the tree would be ordered according to the arc length of their underlying polylines). However, all the other properties of the representation are preserved.

4.4 The Union Operation

The union operations are slightly simpler than the intersection operation. For the union of a P-tree and a P-tree we use a construction similar to the digitization methods for disconncted Ps. The result is a P-tree. Note that the union operation for strip trees is not commutative. Also, we do not define a union operation for a strip tree representing a polyline and a strip tree representing a region. The union of two region strip trees is defined and is a region strip tree. If these two strip trees do not intersect, then the union is straightforward and is identical to the method for polylines. However, if the contrary is true, then we must go to the trouble of defining a new strip tree that represents the union by finding the points of intersection in the same way as was done for region strip tree intersections.

5. Conclusions

Strip trees provide a powerful representation for polylines and areas. Current work is directed towards characterizing their computational complexity more precisely but it can already be shown that the representation is superior to its competitors. The main drawback is that there is a large overhead in terms of space. If n is the required space to represent a polyline then its strip tree will take about 4n space units. Also the creation of a strip tree is a laborious process, requiring 0(n log n) time units. However, neither of these drawbacks are thought to be important in the use of this representation for geographical data bases.

The representation defines strip segments as primitives to cover subsets of the line after [Peucker, 1976]. Our organization of these segments into a tree may be viewed as a particular case of a general strategy of dividing features up and covering them with arbitrary shapes such as depicted by Figure 10. Other attempts in this class have been tried by [Barrow et al.,1977; Burton, 1977; Tanimoto, 1975], but they do not capture the notions of orientation and resolution anywhere nearly as precisely as strip segments, and do not have the union and intersection properties.

Figure 11: The Notion of an Arbitrary Divide-And-Conquer Strategy

Acknowledgements

The author wishes to thank R. Peet and P. Meeker for their work in the preparation of this document. Thanks also go to D. Weaver for his work in implementing the algorithms in SAIL.

References

Barrow, H.G., "Interactive Aids for Cartography and Photo Interpretation" Semiannual Technical Report 12May 1977-11Nov.1977, ARPA contract DAAG 29-76-C-0057, SRI International.

Bently, J.L., "Multidimensional Search Trees Used for Associative Searching" CACM Vol. 18, No. 9, September, 1975.

Burton, W., "Representation of Many-Sided Polygons and Polygonal Lines for Rapid Processing" CACM Vol. 20, No. 3, March, 1977.

Douglas, D.H. and Peucker, T., "Algorithms for the Reduction of the Number of Points Required to Represent a Line or its Caricature," The Canadian Cartographer, Vol 10, No. 2, December, 1973.

Duda, R.O. and P.E. Hart, Pattern Classification and Scene Analysis, Wiley-Interscience 1973.

Minsky, M.L. and S. Papert, Perceptions:an introduction to computational geometry, MIT Press, Cambridge, Mass., 1969.

Peucker, T., "A Theory of the Cartographic Line," International Yearbook of Cartography, 16, 1976.

Sloan, K.R., "Maps and Map Data Structures," forthcoming Technical Report, Computer Science Department, University of Rochester.

Tanimoto, S., and Pavlidis, T., "A Hierarchical Data Structure for Picture Processing" Comp. Graphics and Image Processing, Vol. 4, No. 2, June, 1975.

Turner, K.J., "Computer perception of curved objects using a television camera", Ph.D. thesis, University of Edinburgh, 1974.

FIGURES

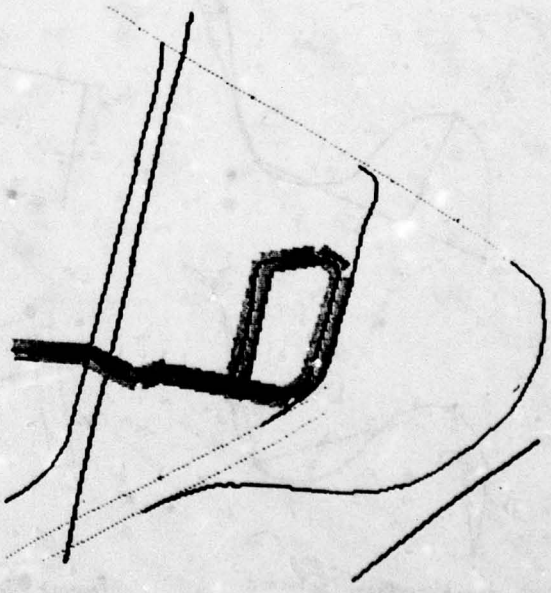


Figure 1. Map features displayed at various resolutions using the hierarchical structure.

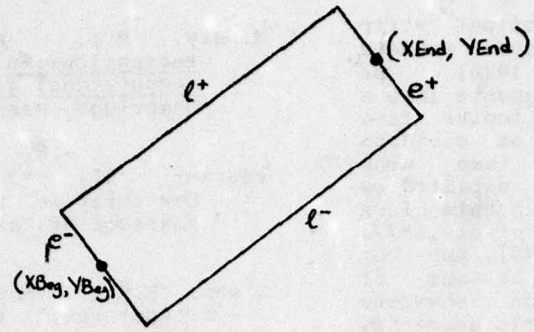


Figure 2: Definition of a Strip Segment.

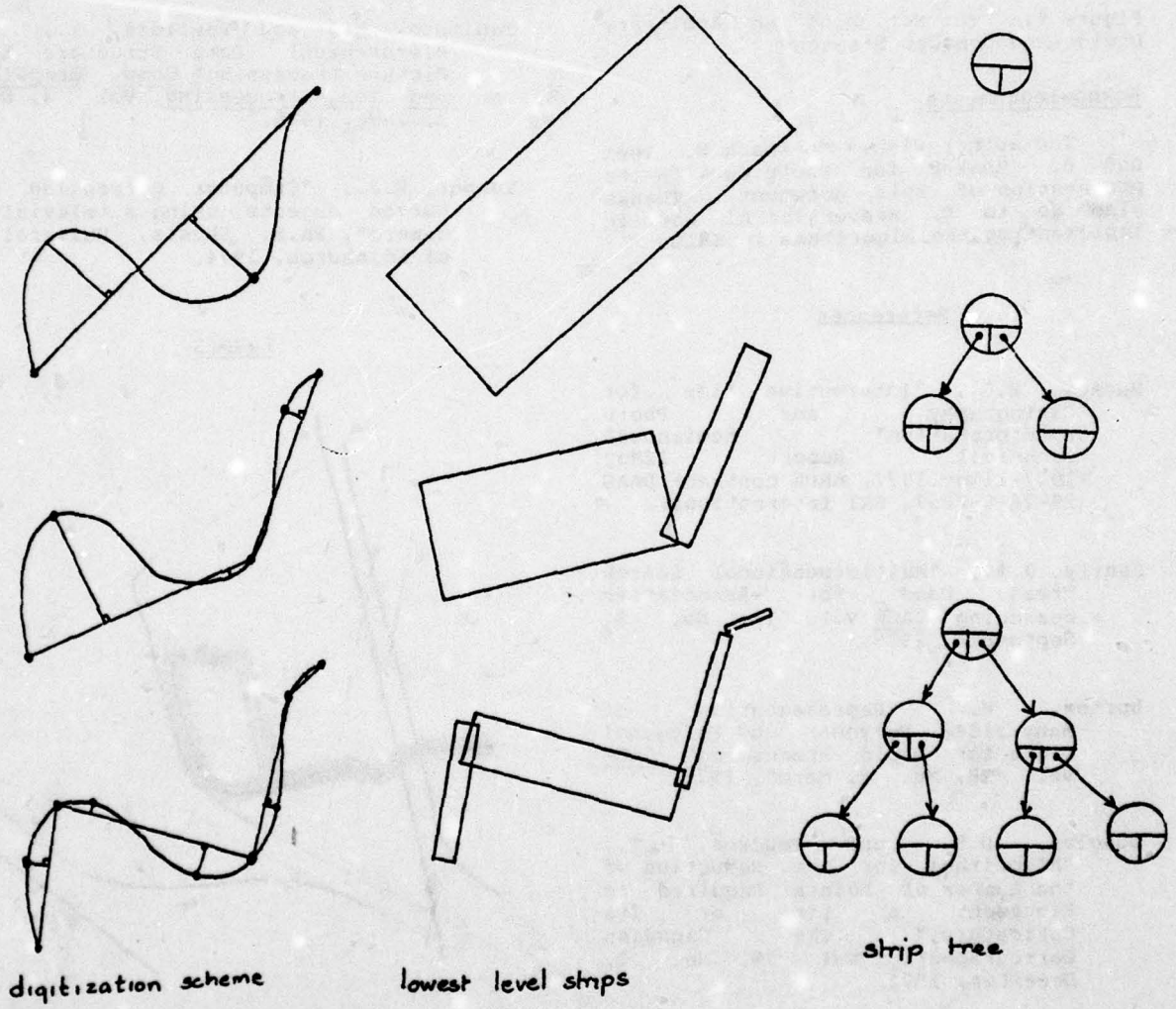


Figure 3: Steps in the Digitization Process.

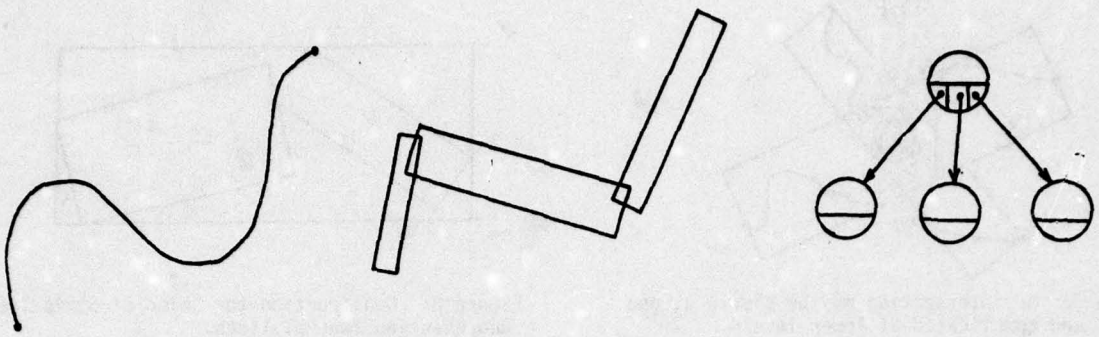


Figure 4. A portion of an encoding using m-ary trees.

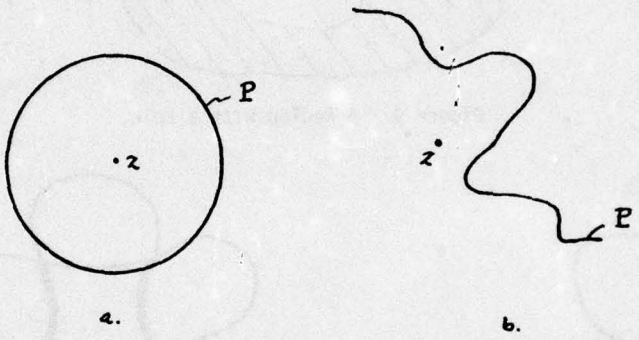


Figure 5: Two of Many Possible Geometrics When Testing the Distance of a Point from a P.

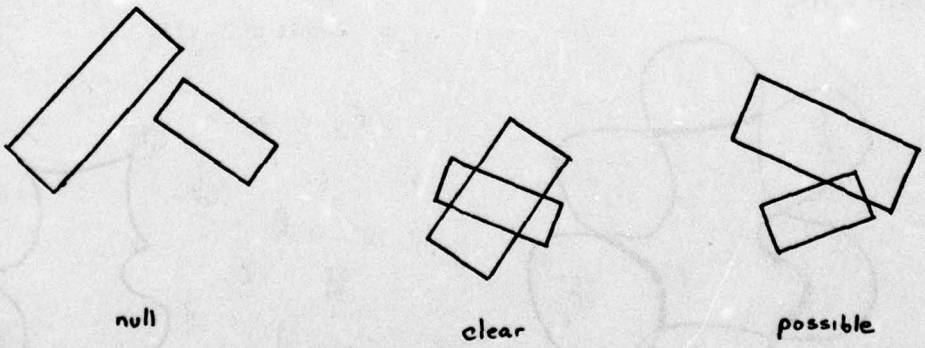


Figure 6: Different Ways Strips Can Intersect

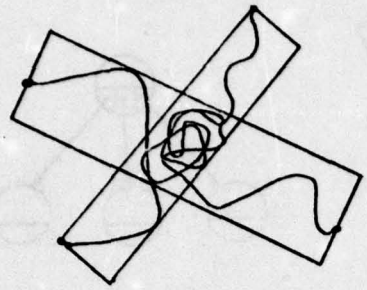


Figure 7: An intersection may be simple at one level and complicated at lower levels.

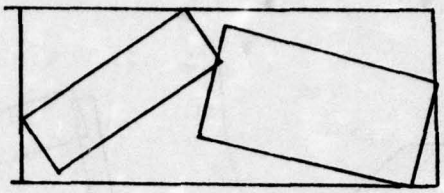


Figure 8: Construction for Union of Strip Trees Representing Two Polylines.

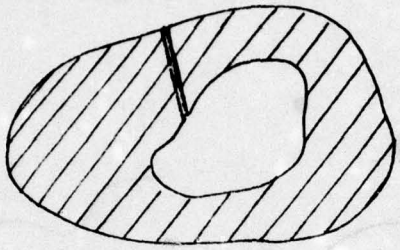
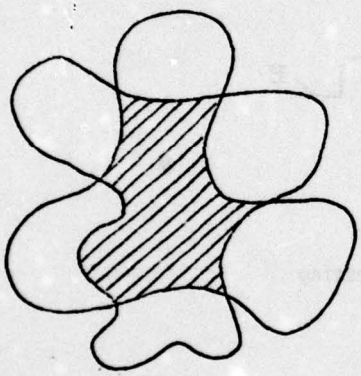
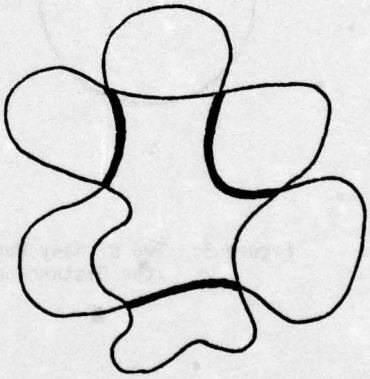


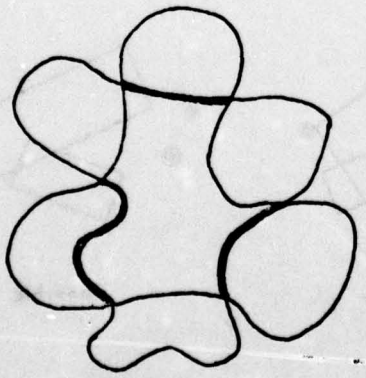
Figure 9: A Region with a Hole.



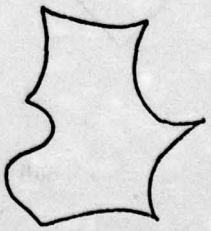
a. Desired Result $A_1 \cap A_2$



b. Result of $T_{L_1} \cap T_{A_2}$



c. Result of $T_{L_2} \cap T_{A_1}$



d. Union of Two Results: the polyline segments covered by the result tree.

Figure 10: Decomposition of Area-Area Intersections

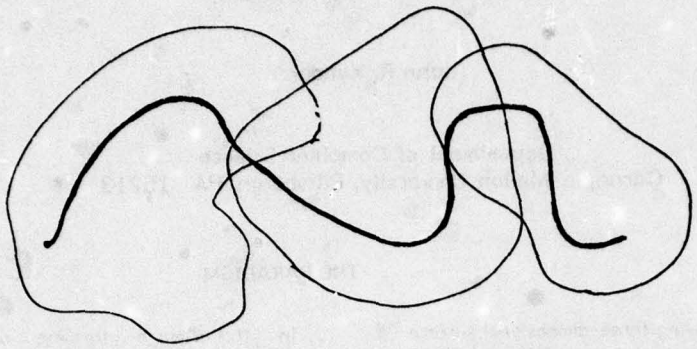


Figure 11: The Notion of an Arbitrary Divide-And-Conquer Strategy

SHAPE FROM TEXTURE: A COMPUTATIONAL PARADIGM

John R. Kender

Department of Computer Science
Carnegie-Mellon University, Pittsburgh, PA 15213

ABSTRACT

A new approach to deriving three-dimensional surface orientation from image textural properties is described. Introduced is a new representational and computational tool, the normalized textural property map, which unites and exploits a large class of low-level image heuristics. An example is given of an application of the paradigm to an (abstract) textured image. Some comments reflect on the relation of this work to existing work on shape.

INTRODUCTION

One central task of image understanding is the recovery of three-dimensional scene information from the two-dimensional perspective transformation that is the image. Generally what is sought is the location and structure of the small number of objects that comprise the typical scene. However, a static, monocular image presents much less to the eye. Objects are visible only as non-occluded opaque surfaces. Surfaces are distinguished by local surface properties (shading, color, texture) subject to deformations due to variations in slant, lighting conditions, relative distance, and other influences. Recovering object descriptions from such a melange of distortions is a formidable task. This paper considers only one surface property (texture) subject to only a small set of imaging phenomena (mainly, the effects of surface orientation).

The approach is necessarily heuristic. Given the infinite number of scenes that can produce identical images, reversing the projection--going from distorted surface properties to three-dimensional surface shapes--requires the judicious use of assumptions about surfaces and the imaging process. The first concern of this paper is what exactly needs to be assumed in order to proceed with the mathematics of analysis. The second, related one is the constraint equations that result.

Most of this paper discusses a new computational paradigm that integrates textural properties, image heuristics, and a representation of surface orientation in a way that allows the quantification of constraints on local surface orientations. The paper also presents, by example, the paradigm at work. The class of relatively model-free constraint equations that result from it are applied to an (abstract) textured scene. Discussion follows. Further results and discussion, as well as much more supporting mathematics, can be found in the author's forthcoming thesis [Kender, 1979].

THE PARADIGM

In the image forming process, surfaces are perspectively projected onto two-dimensional regions of the imaging retina. The images of the texture objects which define the surface are distorted by local surface orientation, relative surface distance, and the characteristics of the imaging device. The task of the visual processor is to deconvolute these effects, but it is only the effects of the last influence, the camera parameters, that are usually known a priori. Recovery of the others depends on simplifying assumptions, based ultimately on intuitive concepts about the physical world. These are the notions of texture "regularity" (both in texture object, and in texture object placement with respect to the surface), and of surface opacity and "smoothness". The general paradigm exploits each of these assumptions in the creation and refinement of the analytic framework described below.

The fundamental conceptual and representational tool is the new one of normalized textural property map. Intuitively, this map relates a given two-dimensional image texel to the small class of three-dimensional texture objects which may have been its source in the scene. More precisely, it is a way of "deprojecting" the effects that surface orientation has on primitive textural properties such as slope in the image, length of major axis of elongation, etc. The map summarizes the answers to the following two-dimensional family of questions: if the microplane underlying this texture object were inclined at this specified three-space orientation, what would the textural property had to have been in the external scene in order to observe the given textural property in the image? Mathematically, a normalized textural property map is a function, whose arguments are the hypothesized surface orientation descriptors, and whose value is the deprojected, "in the scene" normalized textural property. Graphically, the map can be represented as a surface in a three-dimensional coordinate system, with the two-dimensional representation of surface orientation providing the underlying grid, and the normalized textural property providing the vertical axis. It can also be represented as a contour map in the usual way. The contours trace out those microplane orientations giving rise to the same normalized property.

THE NORMALIZED TEXTURAL PROPERTY MAP

The normalized textural property map is composed of four basic ingredients. First, a representation for microplane orientation is chosen. The gradient space

[Huffman, 1971] is one candidate; it represents surface slant by assuming that all microplanes satisfy the equation:

$$-z = px + qy + c$$

This assumes a rectangular coordinate system that aligns the z axis with the line of sight. Surface orientation is then represented by

$$(-\partial z/\partial x, -\partial z/\partial y) = (p, q)$$

which is the gradient, hence the name of the space. Note, however, that other representations for surface slope are possible. For example, it is often the case that representing surface orientations by a spherical coordinate system (the "Gaussian sphere") yields simpler normalized maps.

A second important constituent of the contour map is the considerations of the imaging process. If the image is considered to have resulted from a perspective transformation, the focal length and the central focus point of the imaging apparatus must be known. This complicates the analysis considerably. However, if the image has arisen from an orthographic imaging system, no imaging parameters are required. The focal length becomes infinitely large, and every point in the image can be considered the central focus point. This simplifies the deprojection: the image point (x, y), deprojected onto the plane $-z = px + qy + c$, is $(x, y, -(px + qy + c))$.

In terms of the paradigm, orthography also guarantees that the normalized textural property map is independent of the position of the texel in the image. This is both an advantage and a disadvantage; what is lost is the often powerful constraints perspective deformation imposes on the determination of surface slant. In fact, it can be shown that under orthography some textural properties provide no surface orientation information at all, although they are very useful--and even elegant--under perspective (for example, edge slope [Kender, 1978]). The remainder of this paper will concern only orthographic projection, for simplicity's sake.

The third component in creating the normalized map is the choice of textural property. The following properties, among others, are easily derivable in the image: texel slope, texel length (of a major axis of elongation, say, or of a line element), angle (between texel slopes, or the "T" and "V" joints formed from adjoining larger texels [Maleson, 1977]), areal measures such as area and density (the count of distinguished events, such as edges [Rosenfeld et. al., 1970] or relative extrema [Mitchell et. al., 1977], per unit of image area), and other combinations of the above, such as eccentricity (the ratio of two special lengths, the major and minor axes [Stevens, 1979]), or skewed symmetry (the angle between two special lines, the symmetry axis and the transverse axis [Kanade et. al., 1979]). Each such property generates its own class of normalized map. In general, the maps for the "higher" properties such as eccentricity and skewed symmetry are special cases of simpler ones.

Lastly, to create the map requires the invocation of a general physical world assumption: the uniqueness of the relationship of the texture object to the microplane. That is, a given textural property--say, length--defines the surface associated with it by either lying wholly within it (the texture is "painted"), or by extending above it (it is "pointed"), but it can not do both. (Examples of "pointed" textures are forests of trees, or metropolitan downtowns). This is, of course, a continuum, extending from tangent relations to normal ones, but the analysis is very different at either extreme. The examples that follow will assume assume tangency, the more general case.

These four ingredients come together in the following way. The surface orientation representation provides the coordinate system and the hypothesized scene planes. The choice of textural property defines the texel class that is to be deprojected. The camera model provides the deprojecting function. And the assumption of texture object-to-surface relationship provides further information as to how the various components of the texel are to be deprojected with respect to each other.

As an example, consider the normalized map consisting of the gradient space, the property of length, orthographic projection, and tangency. Let the length element in the image extend from (0, 0) to (L, 0) in the image; its length is L. Assuming tangency allows the deprojection of both ends onto the plane $-z = px + qy + c$ as follows: (0, 0, -c) to (L, 0, -(pL + c)). The deprojected length element has normalized length

$$L_n = L * \text{sqrt}\,(1 + p^2)$$

The contour form of the map is shown in Fig. 1. Due to orthography, this result holds for all equally long length elements oriented parallel to the one given in the example. Further, due to the gradient space, the result for any other slope is simply a rotation of the normalized map to that corresponding direction; see Fig. 2. A catalogue of many such maps for varying ingredients appears in the author's thesis. For example, the map for density under the same assumptions as the above illustration, is

$$D_n = D \,/\, \text{sqrt}\,(1 + p^2 + q^2)$$

which, as expected, is rotationally symmetric.

THE MAP REFINED

The paradigm continues with the two following refinement steps. The normalized textural property map gives no information about surface orientation by itself; it only expresses relationships. It is here that the second and third heuristics about the physical world are invoked. They are the continuity of texture objects ("regularity"), and the continuity of local surface orientations ("smoothness"). In the extreme, these heuristics state that all texture objects in three-space are identical (that is, the texture approaches a "structural" texture), and that all local surfaces are planar. Although not always true, assuming the extremes in small neighborhoods does allow the following mathematical constraints on surface orientation.

Continuity of texture object asserts that neighboring texels must have the same normalized textural property values. All other influences being equal, their maps would be identical. Continuity of local surface orientation similarly asserts that, locally, observed textural properties never differ because of a variations in inclination. What accounts

for dissimilarities is the one degree of freedom left. And that is the relative placement of the texture object with respect to the surface, within the limits of the previous assumption concerning texture object-surface relation. If the assumption was that a texture object lie in the plane, it can still rotate freely in it. Even if the assumption were, say, that it was to be within n^o of the surface normal, many placements lying with the implied cone are possible.

Having assumed away any differences arising from variations in local texture objects and local surface orientations, the normalized textural property maps that result from adjacent texels differ only because of placement effects. This last is removed by intersecting the two maps and finding those microplane orientations that have the same normalized property value. In effect, this mathematical operation of finding all (p, q) so that $P_1(p, q) = P_2(p, q)$ states that only a small set of surface orientations in the scene could have distorted both texture objects simultaneously in such a way that their images can be deprojected into equal normalized values.

As an example, take as texels two length elements as in Figs. 1 and 2. Assume the other constraints as given with the figures. If the length elements appear "nearby" in an image (as in Fig. 3, a very small section of a texture), create and intersect the surfaces of their normalized textural property maps. The result (in Fig. 4), is the the set of possible local surface orientations that could have distorted the assumed three-dimensional regularity (here, equal length) into the given image.

The analysis, of course, can be done strictly mathematically. In the case of two equal length elements the constraint equations, in general, describe an hyperbola. In the above example, it is specified by

$$L_2 * \text{sqrt}(1 + p^2) = L_1 * \text{sqrt}(1 + q^2)$$

where L_i are image measurements. In general, this step of the paradigm ends with a one-dimensional family of plausible surface orientations. It can be further refined by intersecting it with the normalized map of a third nearby texel of the same class. However, often a more useful type of refinement is possible.

THE SECOND REFINEMENT

Suppose the texture is rich enough so that at least one other different textural property can be discovered and assumed to be "regular". Executing the first two steps of the paradigm separately for both textural properties will usually result in one-dimensional locus of allowable surface orientations in both maps. Again invoking the assumptions of continuity of texture object and surface orientation allows these maps to be intersected (or their constraint equations to be mutually solved). That is, although the textural properties (and their maps) may be different, the underlying scenic texture objects that account for both properties are assumed to be physically identical. Under orthography, the analysis usually ends at this stage with a Necker pair of surface orientations (that is, both (p, q) and (-p, -q)). Choosing one or the other of the pair requires the invocation of the last assumption--the uniqueness of surface orientation--in whatever subsequent integrative or relaxation method would be used to recover actual surface depth.

In the example of Fig. 3, assume that the two texture objects in three-space are also mutually perpendicular (that is, their placement orientations are equally offset from the axes of any local coordinate system in the microplane). After the first refinement step, the paradigm generates the constraint equation pq = 0, and a normalized map consisting of the p and q axes themselves. Fig. 5 shows these constraints intersected with the constraints of Fig. 4. The resulting Necker pair of surface orientations indicate the two possible surface orientations for the microplane formed in Fig. 3.

SUMMARY AND COMMENT

The three steps of the surface orientation-from-image texture paradigm are as follows. Step 1: Choose a textural property, find a texel possessing it, and generate its normalized textural property map. Step 2: Choose a similar nearby texel from the same texture, apply Step 1 to it, then intersect the normalize maps: call the resulting map P1. Step 3: Choose a second textural property, apply steps 1 and 2 to get map P2; then intersect maps P1 and P2. Except for certain degenerate cases, this last map is a zero-dimensional set of possible surface orientations for the microplane defined by the local texels. Under perspective, the result is often a unique value.

The normalized textural property map is a close relative to the reflectance map of Horn [Horn, 1977]. Reflectance can be considered as a very primitive textural property. Its image counterpart, pixel brightness, is the most primitive texel possible (a pixel). There is one important difference between normalized maps and reflectance maps, however. A reflectance map assumes knowledge of the (constant) illuminant strength. In the corresponding case of brightness as textural property, the normalized map represents what the unknown illuminant strength would have had to have been; this varies for each surface orientation.

The paradigm given above can be shown to subsume the method of photometric stereo [Woodham, 1978], where the need for differing textural properties is met by varying the scene lighting conditions. Additionally, the paradigm can show that density behaves identically to one restricted case of reflectance. It can therefore add theoretical weight to the heuristic method of blurring texels in order to treat textured regions as shaded regions.

A comment on a different level: much of the paradigm requires the use of heuristic decisions, such as the assumptions that "nearby" textural properties are, in fact, identical. Such assumptions can be expressed as the meta-heuristic: "apparent = true." This finds its expression in the paradigm by such judgements as "nearly equal lengths are equal in three-space, but have been foreshortened," "nearly parallel lines are parallel in three-space, but have been perspectively converged," etc. The meta-heuristic is a restatement of the general view hypothesis, in a form exploitable by this new computational method.

Lastly, the paradigm can throw a little light on the meaning of that obscure object of this study: "texture." It implies that image elements form a texture if they are considered to define a surface and are locally identical in terms of primitive properties. Or, restating, a texture is a collection of image elements whose regularity can be exploited in a computational way to abstract the orientation of their associated surface. Thus, a group of image elements become a "texture" when their properties become regular enough to be used to discover shape.

ACKNOWLEDGEMENTS

I am pleased to acknowledge many helpful and enjoyable discussions with Takeo Kanade and Dave McKeown.

REFERENCES

B. K. P. Horn, "Understanding Image Intensities," Artificial Intelligence, Vol. 8, 1977.

D. A. Huffman, "Impossible Objects as Nonsense Sentences," Machine Intelligence 6, R. Meltzer and D. Michie (eds.), Edinburgh University Press, 1971.

T. Kanade, and J. R. Kender, "Skewed Symmetry: Mapping Image Regularities into Shape," Technical Report, Computer Science Dept., Carnegie-Mellon Univ., 1979 (forthcoming).

J. R. Kender, "Shape from Texture: A Brief Overview and a New Aggregation Transform", Proceedings of the ARPA Image Understanding Workshop, Science Applications Inc., Nov., 1978.

J. R. Kender, "Shape from Texture," Ph.D. Thesis, Computer Science Dept., Carnegie-Mellon Univ., 1979 (forthcoming).

J. T. Maleson, "Understanding Texture in Natural Images," Technical Report, Computer Science Dept., Univ. of Rochester, Nov., 1977.

O. R. Mitchell, C. R. Myers, and W. A. Boyne, "A Max-Min Measure for Image Texture Analysis," IEEE Transactions on Computers, Vol. C-26, April, 1977.

A. Rosenfeld, and E. B. Troy, "Visual Texture Analysis," Proceedings of the UMR-Mervin J. Kelly Communications Conference, University of Missouri, Oct., 1970.

K. A. Stevens, "Surface Perception from Local Analysis of Texture and Contour," Ph.D. Thesis, M.I.T., Feb. 1979.

R. J. Woodham, "Reflectance Map Techniques for Analyzing Surface Defects in Metal Castings," Ph.D. Thesis, M.I.T., June, 1978.

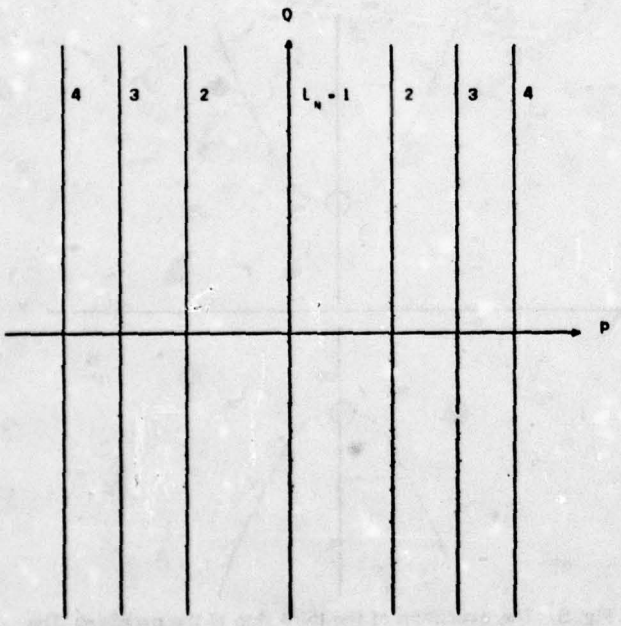


Fig. 1. The normalized textural property map under the conditions of gradient space, unit horizontal length element, orthographic projection, and tangency. The map contours indicate those surface orientations which cause identical perspective distortion; their label indicates the amount.

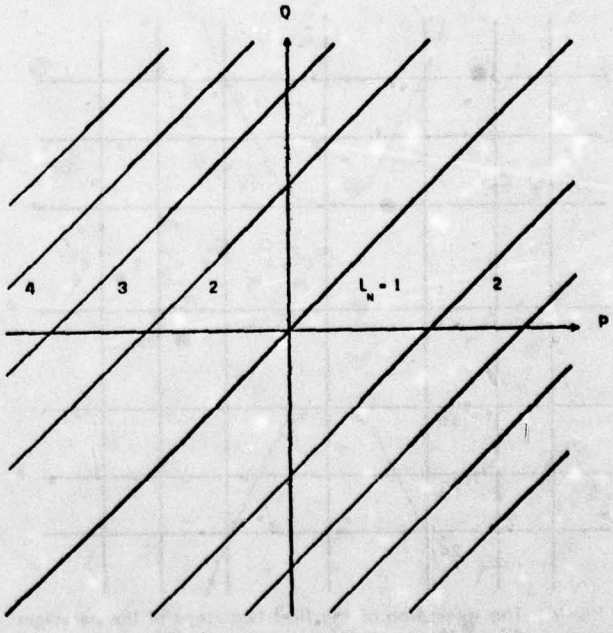


Fig. 2. The normalized textural property map under the conditions of Fig. 1, except that the unit length element is oriented at 45°.

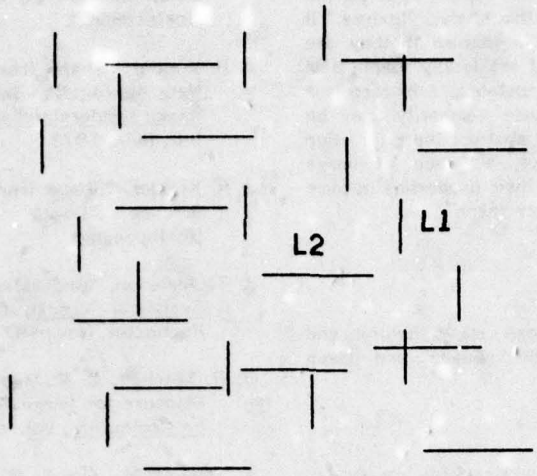


Fig. 3. A simple texture composed of equal length texture objects seen in orthographic perspective.

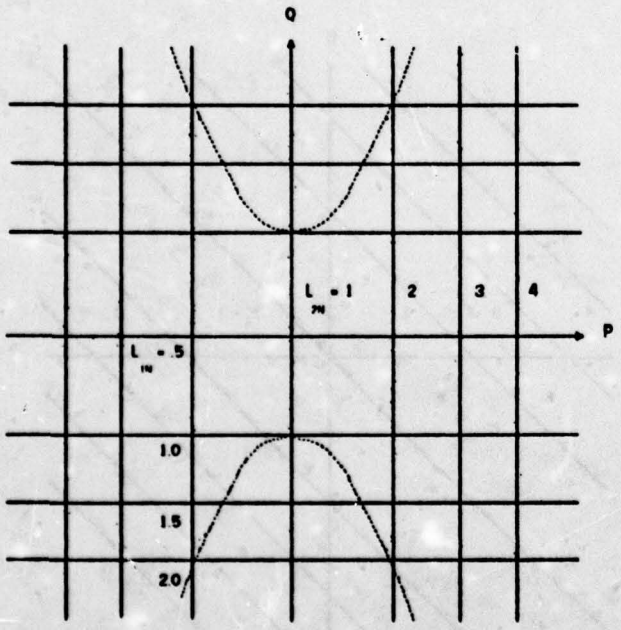


Fig. 4. The execution of the first two steps of the paradigm paradigm on the two labeled texels of Fig. 3. The two normalized textural property maps intersect in a hyperbola. This constrains the possible surface orientations.

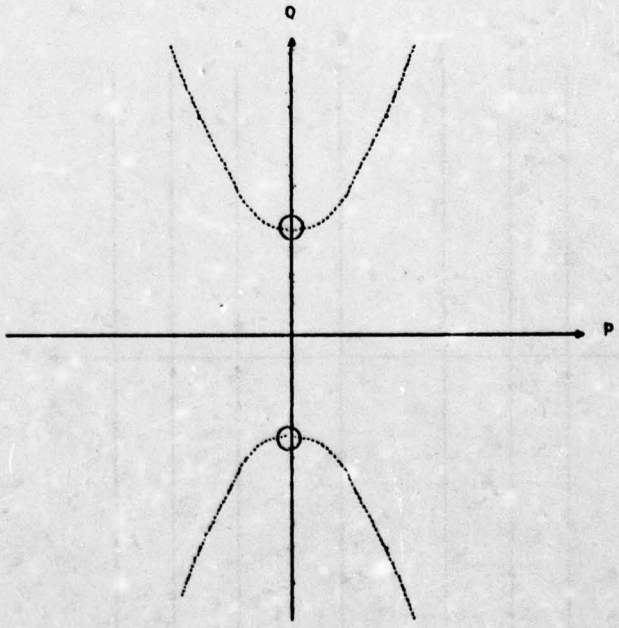


Fig. 5. The execution of the third step of the paradigm. The hyperbola of Fig. 4 is intersected with the degenerate hyperbola consisting of the p and q axes. The latter was obtained by executing the paradigm under the additional assumption that the lengths of Fig. 3 are perpendicular in three-space. A Necker pair of possible surface orientations results.

PROGRESS IN NAVIGATION USING PASSIVELY SENSED IMAGES

Oscar Firschein
Don Gennery
David Milgram
James J. Pearson

Lockheed Palo Alto Research Laboratory,
Department 52-53, Building 204
3251 Hanover Street
Palo Alto, California 94304

ABSTRACT

Our recent work in navigation using passively sensed imagery has concerned the extension of the role of the stereo system, the use of the Night Vision Laboratory terrain model imagery for image velocity sensor experiments, and the study of methods for dealing with vehicle structural flexure effects on the stereo image pair.

INTRODUCTION

The concept of using passively sensed imagery for navigation of a low-flying, slow-speed vehicle was described in Ref. 1. Briefly, an image velocity sensor (IVS) compares sequential image frames to obtain the velocity to altitude ratio, V/H, while a stereo system Ref. 2 computes H, the altitude of the vehicle above the ground, so that the velocity, V, can be obtained. Velocity is also computed using conventional instruments, and an estimate of wind velocity. Dead reckoning navigation, based on a best estimate of wind velocity, is updated periodically with positional corrections made using the passively sensed images.

This paper describes recent activity in the following topic areas, (1) extending the role of the stereo system, (2) the Night Vision Laboratory terrain model and associated imagery, (3) the IVS experiments, and (4) recent stero studies.

EXTENDING THE ROLE OF THE STEREO SYSTEM

The original role of the stereo system was to determine the altitude H, so that the velocity V could be obtained from the V/H measurement obtained by the image velocity sensor. However, as the characteristics of the stereo system were examined, it was noted that there was potential for an expanded role for it. In particular, the system could provide a terrain map in addition to the V/H information, as discussed below.

TERRAIN IMAGE ASPECTS

The stereo system is capable of producing a 2-D terrain image in which relative elevations of image features are given. Thus, we can obtain an image whose pixel values are some fixed positive or negative bias from the actual elevation above sea level. We have been considering ways of matching this type of image against a data base of terrain images in order to obtain positional corrections for the vehicle.

The main advantage of this approach for position verification is the robustness of the derived terrain image compared to images based on scene intensity values. (The derived terrain image should be insensitive to sensor characteristics and lighting conditions, unlike the case of comparing sensed intensity image with reference intensity images).

The following problem areas are therefore currently receiving attention:

1 - Data base construction. Determination of what features can be extracted from a topographic image, and the data compression and smoothing techniques available.

2 - Topographic image determination. At present, to solve for the camera model one finds corresponding regions in a pair of images. We are examining whether feature-based correspondence offers any advantages.

3 - Terrain matching. There are several approaches to terrain matching being considered. In addition to the 1-D TERCOM approach, there are the following 2-D alternatives:

- area correlation based on terrain elevation values rather than scene intensity.
- spatial matching of terrain features, using techniques such as chamfer matching, array relaxation, minimal spanning tree matching, or Voronoi decompositions.
- symbolic description matching, using techniques such as M* algorithms, graph relaxation, and dynamic programming.

USING STEREO FOR V/H MEASUREMENT

The original plan for measuring velocity from imagery called for height (H) to be measured by high-resolution stereo and the ratio of velocity to height (V/H) to be measured by the IVS. However, on a vehicle with no inertial reference system, the IVS is subject to errors caused by changes in attitude of the vehicle. A charge in attitude causes a shift in the image which will be confused with a shift caused by linear motion, causing a false reading of V/H to be obtained.

Therefore, a different way of obtaining the V/H information has been investigated. This method uses stereo with a baseline produced by vehicle motion, using a single camera. The stereo camera model solver can compute the change in vehicle attitude. Then the result of the stereo computation is the ratio of height to the length of the stereo baseline (4), that is H/R. Combining this with H from high-resolution stereo as before produces R, that is, the distance that the vehicle moved between pictures. Furthermore, the stereo camera model gives the direction of this motion relative to the camera coordinate system, so that the vector motion is known. Dividing this by the time interval produces velocity.

If the camera-cluster method of calibrating the camera model for the high resolution stereo is used, as described below, one of these same camera clusters can be used for the stereo needed for the H/R computation. The exposures would have to be very close in time, say about ten feet of vehicle motion, so that some of the same points will still be in the fields of view of the narrow-angle cameras. The camera model can be very accurately calibrated because of the high-resolution of these cameras and the wide field of view that they collectively span, so that an accurate H/R can be computed with the short ten-foot baseline. These considerations are the same as those discussed below in connection with obtaining H by this method.

No matter what method of calibrating the high-resolution stereo is used, a single wide-angle camera can be used to obtain the stereo for the H/R measurement. This could be the same camera used to obtain stereo for map-matching or to obtain non-stereo information. Because of the poorer angular resolution of the wide-angle camera, the baseline would have to be longer (a few hundred feet). However, this is no problem as long as there is sufficient overlap between views and sufficiently low distortion, because the baseline is generated by vehicle motion. The accuracy that can be obtained by this method is given in Appendix A.

IMAGERY FROM THE NIGHT VISION LABORATORY TERRAIN MODEL

The Night Vision Terrain model is a physical model of terrain at a scale of 400:1. A detailed topographic map of the model is available from NVL, Ref. 4. A gantry crane arrangement traveling over the model is used to obtain video imagery. Although at present, movement of the crane and the camera orientations are set manually, a future enhancement will allow both the gantry and the camera to be moved under computer control.

This model offers a source of terrain imagery controllable with respect to:

- flight path
- illumination
- camera orientation
- type of camera
- selection of terrain type (cultural or rural features)

Selected frames from a video tape of several passes over the terrain map have been digitized and used for image velocity sensor experiments (see below) and will be used for stereo experiments to derive topographic images. Two typical scenes are shown in Figures 1 and 2, and the associated topographic map for the region is shown in Fig. 3. Initial stereo results using a NVL image pair are given in Appendix B.

IMAGE VELOCITY SENSOR EXPERIMENTS

A series of experiments using the image velocity sensor on the region shown in Fig. 1 were conducted. A pair of 128 x 128 regions were extracted from two 512 x 512 frames and processed using the special IVS processor.

It was found that the camera was forward pointing and not aimed along the flight path. This resulted in large perspective effects in the images; from frame to frame, objects at the top of an image moved less than those at the bottom. In addition, objects on the left side of the image moved parallel to the direction of motion, while those on the right side translated from left to right. As shown in Fig. 4, the IVS displacements depended on the portion of the image from which the 128 x 128 subregion was selected. This effect occurred even for images having 80% overlap.

Thus, we found that if the vehicle pitch and roll are large enough to cause perspective effects, the IVS should use a 128 x 128 image covering a large field of view to compensate somewhat for the offset errors. This was verified by compressing the 512 x 512 image (by sampling to 128 x 128) and running the experiment again.

STRUCTURAL FLEXURE COMPENSATION

In order to determine vehicle altitude using stereo, we cannot use two looks of a single sensor because we do not know the baseline to be traveled. Instead, we must use two cameras, mounted on opposite wings or mounted fore and the aft on the fuselage. If we are dealing with flight altitudes on the order of 1000 feet and cameras that are at least 10 feet apart, a 1% accuracy in altitude determination requires a resolution of 0.1 miliradians. When dealing with this accuracy in resolution, the relative camera orientation change due to vehicle structural flexure becomes important. Various techniques for measuring the camera-relative orientations were investigated, including the sensor cluster approach shown in Fig. 5.

In the cluster calibration approach, a cluster of sensors having a narrow field of view is used at each position, as shown in Fig. 5. A series of experiments was made for the various cluster forms shown in Fig. 5 , and the following table shows the resulting pan accuracy due to the

uncertainty in the point group positions, and the pan error resulting from a scale factor error of 1 part in 1000 in the point spacings. (With single cameras, this would correspond to an error in one focal length of 1 part in 1000). The values are in degrees.

Point Group	Pan Error From Points	Pan Error From Scale
3A	.0109	.058
3B	.0109	.058
4A	.0091	.055
4B	.0029	.00057
5	.0029	.00057

The roll error in all of the above cases was less than .02 degree. In Ref. 3 we determined that a maximum pan error on the order of .0057 degree and a maximum roll error on the order of .29 degree, was allowable. We can therefore see that the last two cases are satisfactory, whereas the first three are unsatisfactory with the assumed scale factor accuracy and would still be questionable even if the scale were known exactly. Therefore, it seems safe to assume that at least four points will be needed with this method. Furthermore, the point or points for the main height measurement should be directly under the cameras. Since Group 4B does not contain such a point, an extra pair of cameras would seem to be required with it, and its results might as well be used in the camera model determination. Therefore, Group 5 apparently is the best arrangement to use, since it contains such a point. The total effect on height accuracy of the camera model determined from Group 5 is 5.0 feet. Combined with the point accuracy of 10 feet, this produces a total accuracy of 11.2 feet.

REFERENCES

1. O. Firschein and J. J. Pearson, "Artifical Intelligence Concepts Applied to Navigation Using Passively Sensed Images", Proc.: Image Understanding Workshop, Pittsburgh, Penna, November 14-15, 1978.

2. D. B. Gennery, "A Stereo Vision System", Proc: Image Understanding Workshop, Palo Alto, CA, Oct. 20-21, 1977.

3. Passive Navigation R&D Status Report No. 2, 15 Sept 1978. Lockheed Palo Alto Research Lab.

4. Night Vision Laboratory Terrain Model, Defense Mapping Agency Topographic Center, Washington, D.C., 1978.

APPENDIX A

ACCURACY STUDIES OF STEREO V/H MEASUREMENT

A computer simulation was done to test the accuracy of calibrating the stereo camera model when using a single wide angle camera. The following assumptions are made: The height is 1000 feet, each image is 500 pixels by 500 pixels, and the accuracy of the stereo matches is one pixel. A nominal camera model was assumed, with azimuth equal to 90 degrees, and the other four angles all zero. Several cases were considered: 80% overlap between the two pictures, with five points, one in the center and one 50 pixels from each corner of the common region, as shown in Figure A-1; and points spaced every 100 pixels over the common region, with overlaps of 80%, 60%, and 40%, producing 20, 15, and 10 points, respectively, as shown in Figure A-2. Furthermore, for the cases with 80% overlap, two different camera fields of view were assumed: a focal length of 500 times the pixel spacing, corresponding to a field of view of 53 degrees by 53 degrees (1000 feet by 1000 feet on the ground); and a focal length of 250 times the pixel spacing, corresponding to a field of view of 90 degrees by 90 degrees (2000 feet by 2000 feet on the ground). For the cases with 60% and 40% overlap, only the 500-pixel focal length was used (1000-foot view on the ground). From the above assumptions, it can be derived that the length of the stereo baseline (separation of the two camera positions) has the following values: 400 feet for the cases with the 250-pixel focal length (80% overlap); and 200, 400, or 600 feet for the cases with the 500-pixel focal length, according to whether the overlap is 80%, 60%, or 40%, respectively.

For each of the above cases, a camera model solution was done using the points in the overlapping region, and the accuracy of the resulting camera model was propagated into the value of height computed using this camera model, for points approximately in the center of the field of view. The resulting height accuracy (considering camera model effects only) for each of these cases is shown in the following table.

Fig	Overlap %	Points	View Ft	Baseline Ft	Height Accuracy Ft
1	80	5	2000	400	10
2	80	20	2000	400	9
1	80	5	1000	200	42
2	80	20	1000	200	36
2	60	15	1000	400	28
2	40	10	1000	600	38

Since we are looking for an accuracy on the order of 1% (10 feet at an altitude of 1000 feet), only the first two cases in the above table are satisfactory. (This comment applies only to the H/R measurement being discussed here. For stereo used for map matching, the relative accuracy rather than the absolute accuracy is more important, and the camera model effects are not very important, since they tend to affect nearby points almost equally. The accuracy considering only individual point errors is 10 feet or less for all of the above cases). However, these first two cases require a wide field of view (90 degrees), and thus may not be practical.

In order to achieve the desired accuracy with the 53-degree field of view, two possibilities present themselves. One is that the accuracy of the points may be better than the assumed one pixel. With a good signal-to-noise ratio in the

pictures it is possible to achieve a standard deviation of .289 pixel (or even better if interpolation is used). Multiplying by this factor shows that the fifth case in the table would be less than 10 feet, and none of the cases would be very far over 10 feet. However, experiments must be done with real images of typical terrain to determine whether this accuracy can be consistently obtained. The other possibility is to use a greater number of points. With a given distribution of points, the standard deviation of the result is inversely proportional to the square root of the number of points. Therefore, instead of the third case in the table, suppose that each of the five points is replaced by a cluster of 25 points, so that there are 125 points in all. (This could be done by constraining the interest operator to look for points only in the 100-by-100 areas of the picture centered on these points). The fact that there is some spread in these clusters around the original points would produce a slight additional improvement in accuracy (provided that the cluster is centered on the original point), but if this fact is neglected, the accuracy of height would be five times as good, or about 8 feet. Again, experiments must be done with real images or typical terrain to determine whether the interest operator can reliably find good points this closely packed.

An important additional point must be mentioned. That is, the above analysis assumes that there is no systematic error in the position of the points, such as would be caused by camera distortion. Appreciable distortion would introduce a large absolute error into the height that would make this method completely unsuitable for the purpose being discussed. (Unless the distortion is large, the method would still be useful for other stereo purposes, however, where only relative accuracy is important). Thus a solid-state TV camera with a lens whose distortion is accurately known would probably be required. Any type of scanning using the deflection of an electron beam would produce variable distortions that would probably be too large.

APPENDIX B

STEREO RUN USING NIGHT VISION LABORATORY IMAGE PAIR

The Stanford stereo system, Ref. 2, has been used to compute the camera models on a pair of Night Vision Laboratory images. Of particular significance in this initial effort was the number of points that the "interest operator" in the program would find, and whether the camera parameters would be found successfully.

The two images used are shown in Figs. B-1, B-2, and the points of interest found are shown in Figs. B-3, B-4, with each interest point marked with an arbitrary code. At least 50 points of interest were found, with good spatial coverage, and there was no problem with ambiguous points. In Fig. B-5 the points of interest for each image have been superimposed, and a line drawn between some of them to indicate the motion of the points. Note that points of the left side of Fig. B-5 have moved almost vertically (e.g. points 2 and =); while the points on the right (such as * and R) also exhibit a movement to the right.

The computed camera parameters and the associated standard deviations are shown in Table B-1. Note that the pan, tilt, and roll are small, as are their standard deviations. While the camera conditions under which the images were taken are not available, the results for elevation, about 65 degrees down from the horizontal, and azimuth, pointing to the right of the line of motion, do agree with human observation.

The experiments will be continued to obtain the terrain elevation image for this pair of images. The terrain image will then be compared with the NVL terrain map. Further efforts will concentrate on feature analysis.

Table B-1

Camera Parameters Obtained

	Angle (Degrees)	Standard Deviation (Degrees)
Azimuth	-14.8	3.4
Elevation	64.7	1.2
Pan	-0.3	0.2
Tilt	-0.6	0.4
Roll	0.3	0.1

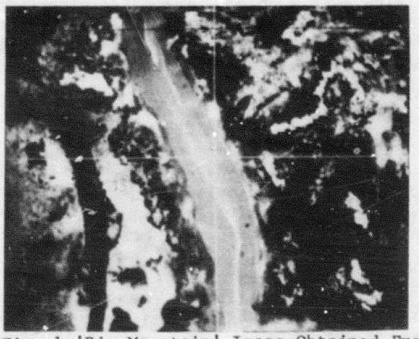


Fig. 1 'Big Mountain' Image Obtained From the NVL Terrain Model

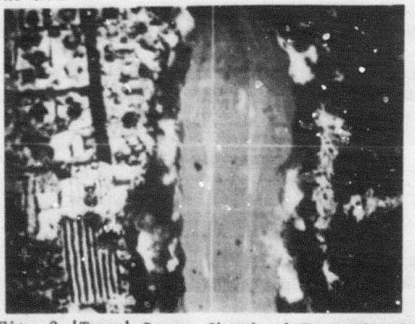


Fig. 2 'Town' Image Obtained From the NVL Terrain Model

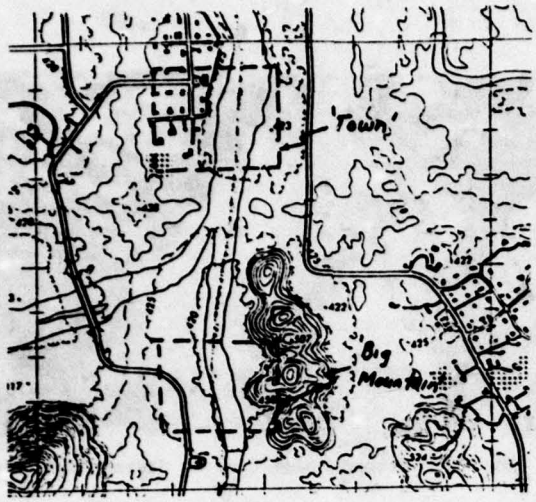


Fig. 3 Portion of Topographic Map of NVL Terrain Model

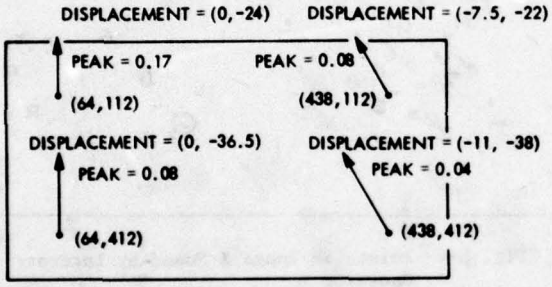


Fig. 4 Image Velocity Sensor Offsets Obtained from Various Portions of the Image

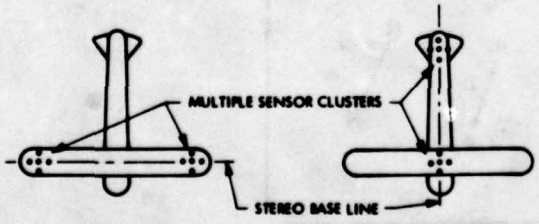


Fig. 5 Location of Sensor Cluster on Vehicle

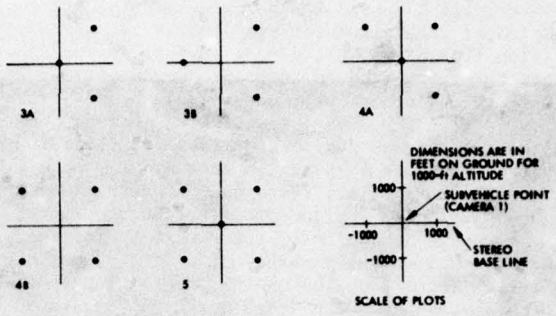


Fig. 6 Some Sensor Clusters Investigated

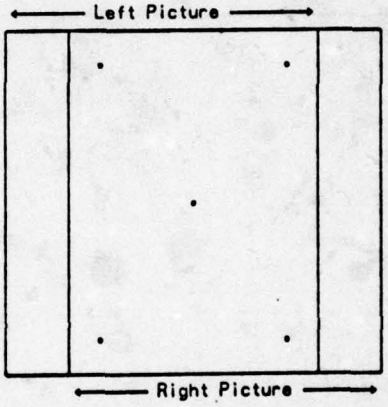


Fig. A-1 Case of 80% Overlap

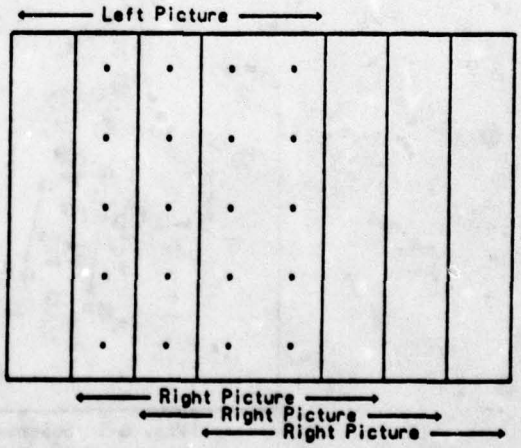


Fig. A-2 Cases of 40%,60%, and 80% Overlap

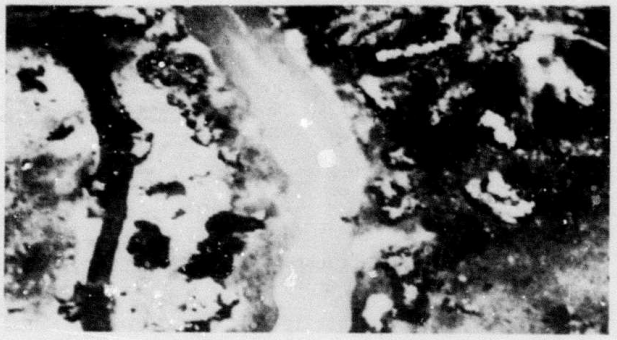


Fig. B-1 First Image Used in Stereo Experiment

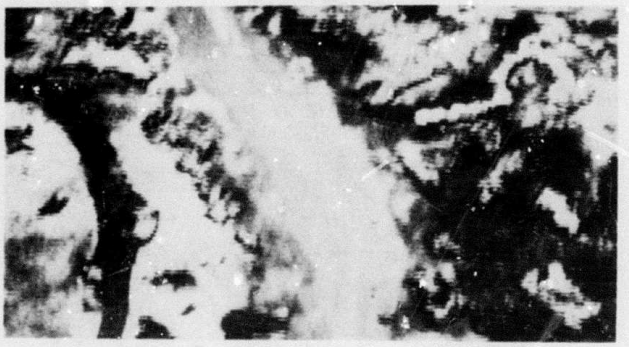


Fig. B-2 Second Image Used in Stereo Experiment

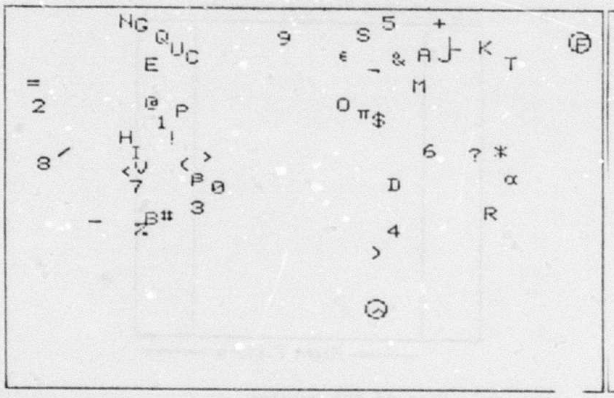


Fig. B-3 Points in Image 1 Found by Interest Operator

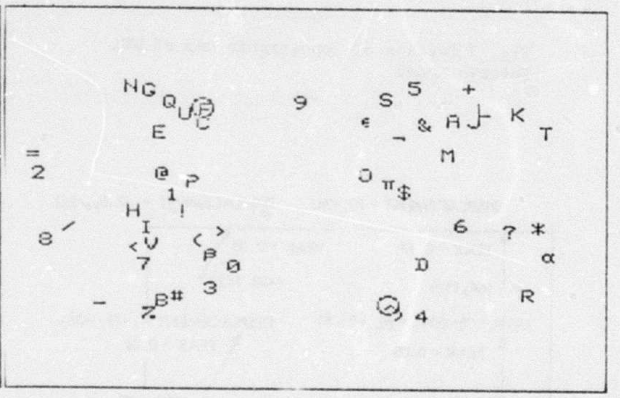


Fig. B-4 Points in Image 2 Found by Interest Operator

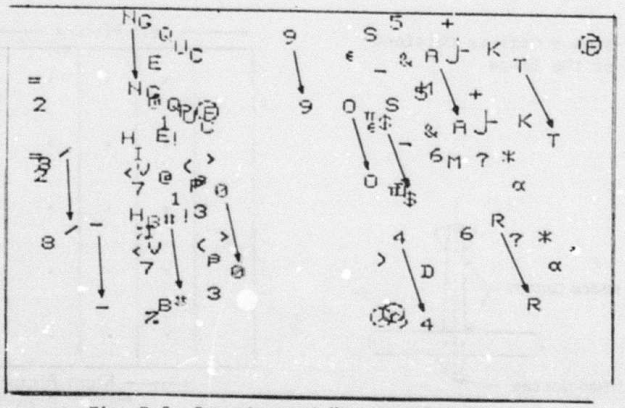


Fig. B-5 Superimposed "Interest" Points, Showing Point Motion Between Images. (Note: Circled Points Were Considered by the Process to Have Been Mismatched, and Were Excluded from Further Analysis).

CONTEXT DEPENDENT AUTOMATIC IMAGE RECOGNITION SYSTEM

Raj K. Aggarwal, Durga P. Panda,
Michael E. Bazakos, Tim Wittenburg

Honeywell, Inc.
2600 Ridgway Parkway
Minneapolis, MN 55413

ABSTRACT

Over the past two years Honeywell has developed a context dependent automatic image recognition system for analyzing the imagery automatically and detecting tactical as well as strategic targets in the image. The main features of the image recognition system are sequential frame processing, symbolic image segmentation, syntactic recognition, recognition of multicomponent objects and conflict removal. In this paper we describe various components of this context dependent automatic image recognition system and information flow between these components.

INTRODUCTION

A general block diagram of the context dependent system is shown in Figure 1. The image is first segmented by two complementary segmentation schemes. Next, man made object (MMO) is detected in the segmented image by a statistical technique. The output of the MMO detector is processed by secondary screening target detector which further reduces false alarms based upon true size, temperature, etc., of the targets on the ground plane. Sequential frame analysis is used to improve the performance of the target detector. A syntactic recognition scheme uses knowledge of the structural description of the targets in recognizing targets that are large enough to show structural detail. For images that are too small to show any structural detail a statistical recognition scheme is used. Sequential frame analysis is employed to take advantage of frame to frame consistency in the imagery to improve the overall performance of the system. Extended objects such as river, road, bridge, highway, etc., are classified by the background classifier. The outputs of the background classifier, the small image statistical classifier and the large image syntactic classifier are combined by a configuration analysis scheme to recognize multiple component structures such as SAM sites, vehicle convoys, airport and to remove conflicts. An example of conflict removal is using the fact that wheeled vehicles are not found in the middle of lakes or rivers. The output of the conflict removal function is recognized targets that have tactical importance or are important based on mission analysis. In the following sections we describe the individual components of the system in detail.

SEGMENTATION

In picture recognition problems, the segmentation task can be divided into two classes: texture based and non-texture based segmentations. We use prototype similarity transformation technique[1] for non-texture based segmentation. It is a method for transforming an image into a set of symbols, each of which represents the relationship of a local region to other parts of the image. A general block diagram of prototype similarity transformation is shown in Figure 2. Generating prototypes is equivalent to finding a maximal set of mutually dissimilar cells. A cell is a pixel or a collection of pixels, depending upon the required resolution in the segmented scene. The generated set of prototypes is used to label each cell in the image. *A priori* information about the scene is used to guide an inference process to give meaning to each cell in the symbolic image.

Segmentation of individual components of a target can also be done by using the prototype similarity transformation technique. This is done by iterative use of the technique at progressively higher cell resolution as shown in Figure 3.

Texture-based segmentation scheme uses a two-dimensional difference histogram. The difference histogram is similar to the co-occurrence matrix of Harralick[2]. The histogram gives a measure of the joint probability density function of gray level and gray level difference pairs. Points in a window around each mode or local maximum in this histogram correspond, ideally speaking, to a region with a particular texture pattern in the input image. Given a mode location in the two-dimensional histogram $M(I, \Delta I)$ the window around it is selected by locating nearest valleys or local minima around the peak. The output of the region extraction operation, which extracts regions in the image corresponding to various modes in the histogram, is a set of scattered pixels corresponding to each region. A noise cleaning method, called "expand and shrink", is used for cleaning the segmented images based on connectivity.

SEQUENTIAL FRAME ANALYSIS IN IMAGE SEGMENTATION

In both the segmentation techniques results of the segmentation of previous frame are used as the starting points of segmentation in the present frame. In prototype similarity transformation the initial choice of prototypes is the same as the prototypes generated in the previous frame. The advantage of this is that the performance of the segmentation technique approaches the asymptotic value as time proceeds. In the difference histogram technique the average intensity I and intensity difference ΔI in each of the regions segmented in the previous frame are used as the center points of the difference histogram windows. This has the added advantage of reduced computational requirement because of reduced search time in the difference histogram.

TARGET DETECTION

The output of non-texture segmentation is used to detect and recognize targets such as tanks, trucks, and APCs. A preliminary screening of non man-made objects (MMO) is first performed on the segmented image by a linear classifier. Table 1 shows features used in classifying the segmented objects into MMO vs non-MMO.

Table 1: Features for MMO Detection

Number	Feature
1	Number of Scan Lines
2	Area
3	Edge Straightness
4	Max (Width/Length, Length/Width)
5	Edge Discontinuity
6	Number of Edges
7	Number of Brights
8	Position in the Initial Scan
9	Position in the Final Scan
10	Final Scan Line

The current performance of the classifier is over 90% detection at 4% false alarm.

Secondary Screening--The detected objects are further screened based on the true size, temperature, or other physical properties. Classification for secondary screening is performed using image features, sensor parameters, and physical dimensions of all anticipated targets. The sensor parameters needed are the angular subtense of the Field of View (FOV), (ξ, η), pixel dimensions of the FOV in the image plane, (M, N), the angle of depression of the LOS, α, and the altitude of the sensor location or carrying aircraft, H. Figure 4 shows the performance of the target detector for AAD-5 and FLIR sensors under various simulation conditions.

SEQUENTIAL FRAME ANALYSIS FOR TARGET DETECTION

System noise in an image recognition system affects the performance of the system in two ways. Firstly, the target may fail to meet the segmentation criteria of the system, resulting in a missed target. Secondly, the feature values of the segmented objects may be erroneous, resulting in missed targets as well as false alarms. Improved false alarm and detection is achieved by accumulating information regarding the locations and the feature values of the objects from frame to frame.

In the sequential frame analysis we first determine an interframe sequence of extracted objects containing a given candidate target in the present frame. We then determine if the classifier result on the candidate target in the present frame is consistent in certain manner with the classifier results on other objects from the past frames in the sequence. An inconsistent classifier result is modified in some prespecified manner that yields better classification result. This method of "smoothing" the classifier result consists of three distinct steps, frame alignment, interframe object matching, and decision smoothing.

The frame alignment technique estimates the relative translation, rotation, and scale change between two successive frames. To estimate this frame-to frame change, segmented image frames and an associated feature vector for each segmented object in the frame is used. The two frames are then aligned with each other by performing appropriate transformation. A symbolic matching of segmented objects in the two frames is then performed to determine the correspondence between objects in the successive frames. The classifier decision made on a candidate target in the present frame is modified based on the decisions made the same object in the immediate past frames using maximum likelihood estimate.

TARGET RECOGNITION

At short ranges, when the target images are large enough to show detailed structure, linguistic recognition techniques are used to classify the detected targets into one of three target types: tank, truck, and APC. When the target image is too small to show any structural detail, a knn classifier is used to classify the targets. The features used for statistical classification of small image targets are the detection features (Table 1) and intensity and boundary moments.

As it turns out, the number of features required for statistical pattern recognition is often very large, which makes the idea of describing complex patterns in terms of a (hierarchical) composition of simpler subpatterns very attractive. Also, the number of possible descriptions is very large in the case of tactical targets from relatively close range. In such a case it is

impractical to regard each description as defining a class. Consequently, the requirement of recognition is better satisfied by a syntactic description of each class rather than by its classification.

The assumption in this syntactic approach to tactical target recognition are:

- Images of tactical targets are large enough to show structure.
- It is easier to recognize target components than the target.

The first assumption deals with the sensor-target range. If the range is too large to show any details inside the target, one would have to resort to statistical recognition techniques. But as the sensor-target range decreases and the target structure becomes discernable, syntactic recognition schemes become feasible. From our experience, if the target area is of the order of one-half to one percent of sensor FOV, syntactic recognition schemes are feasible. This translates to about a ten centimeter pixel resolution.

The second assumption deals with the relative ease of recognizing target and its components. If it is easier to recognize a target than its components, as would be the case when target image is only a few pixels, one would not employ syntactic recognition schemes. But in low quality images where the recognition based on target outline is not very reliable, a syntactic scheme can be successfully used to recognize targets provided the assumption on target image size holds. Even for good quality images, target orientations will result in different target outlines. Consequently, one will need several classifiers for each type of target. In principle, one set of syntactic rules can be generated to recognize the target from all aspect angles. Syntactic recognition schemes can also be successfully used for partially occluded targets where conceivably statistical recognition schemes would fail.

We have developed and applied a syntactic classifier [3] to a training set of 27 FLIR Images containing tanks and trucks. We obtained 100% recognition at zero false alarm. The technique is still to be tested on a large data set.

BACKGROUND CLASSIFIER

Recognition of extended object is done by nearest neighbor (NN) rule using 9 features: length, average width, area, height, average intensity, maximum intensity, minimum intensity, average contrast and peak contrast. We are currently developing an optimal hierarchical sub-grouping of the features for the purpose of minimizing computational complexity. The objective is to retain only a subset of prototype samples for each sub-group.

CONFLICT REMOVAL

Conflict removal combines object information and relational context information for modifying classifier decisions that are inconsistent with our world knowledge. The process requires modeling and representing the world knowledge regarding objects in the scene and determining an optimal way, called search strategy, of examining the scene using the knowledge model. The method can also be used for recognizing scene components containing multiple objects. Examples of such objects are airports, SAM sites, convoys, and bunkers. Various methods of modeling the knowledge and using the model to recognize mission oriented scenes exist in the literature[4]. The methods depend on the particular application of the system. We have combined appropriate concepts from various systems and developed a knowledge model and a search strategy for military tactical importance in imagery[5].

Conflict removal is performed by detecting inconsistent configurations in the scene. An example is a tank in the middle of a river. If the structural relationship between two recognized objects, one recognized as a tank and another recognized by the background classifier as a river, is such that the tank is located in the middle of the river then that particular configuration is flagged as inconsistent with the world knowledge network model. In such cases the target, the tank in our example, is reclassified to a "don't know" category. This conflict or inconsistency is removed by sequential frame analysis, which is analogous to a human operator taking several looks at the scene of interest when he is not confident of his recognition result for the given scene.

SYSTEM SIMULATION

Until the present time we have simulated, optimized, and tested components of the context depend automatic image recognition system individually. We have initiated the task of simulating the entire system as a whole and evaluating the performance of the system while characterizing the impact of the performance of one component on other components in the system. We have identified a test data base for the purpose. To help speed up the system simulation we will use the I^{2}S Model 70 video data processing system recently acquired and interfaced with the Level 6/43 computing system of the Image Processing Laboratory of the System and Research Center. Special architecture of the Model 70 system will enable us to simulate parallel processing of certain functions such as prototype similarity segmentation and texture based segmentation. This will help us characterize the need for digital parallel processing in an advanced automatic image recognition system.

ACKNOWLEDGMENTS

The authors wish to thank Dr. M. Geokezas, the Section Chief of Signal and Image Processing at Honeywell for helpful discussions and encouragement. They also wish to thank Ms. Betty Tremel for skillful typing in preparing this manuscript.

Reference

1. R. K. Aggarwal, Image Segmentation Using Prototype Similarity, Proceedings of the IEEE Conference on Pattern Recognition and Image Processing, May 1978.

2. R. M. Haralick, K. Shanmugan, and I. Dinstein, Textural Features for Image Classification, IEEE Transactions on System, Man and Cybernetics, Volume SMC-3, November 1973.

3. R. K. Aggarwal and T. M. Wittenburg, Syntactic Recognition of Tactical Targets, Proceedings of the DARPA Image Understanding Workshop, November 1978.

4. F. Hayes-Roth, Representation of structured events and efficient procedure for their recognition, Pattern Recognition, Volume 8, 1976.

5. D. P. Panda, Configuration Analysis for Scene Recognition, Technical Memo, Honeywell, Inc., Systems and Research Center, March 1, 1979.

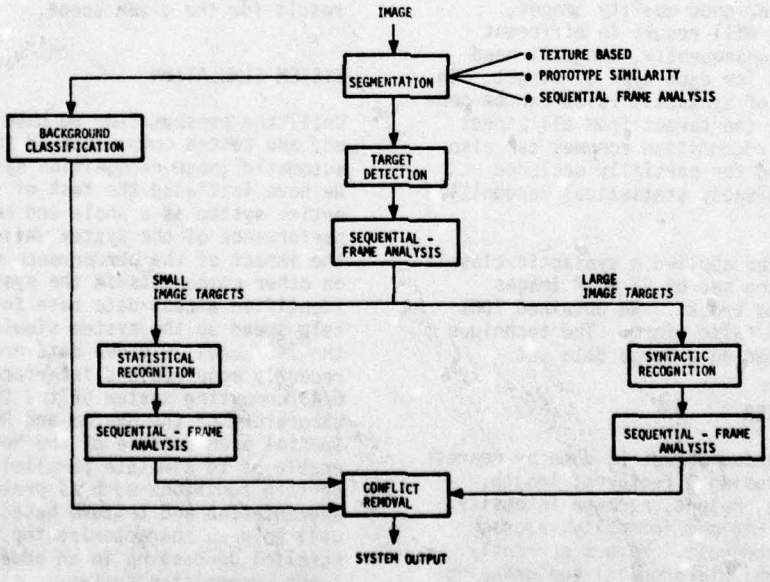


Figure 1. Context Dependent Automatic Image Recognition System

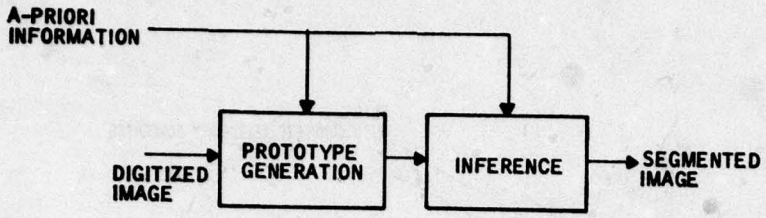


Figure 2. Block Diagram for Prototype Similarity Transformation

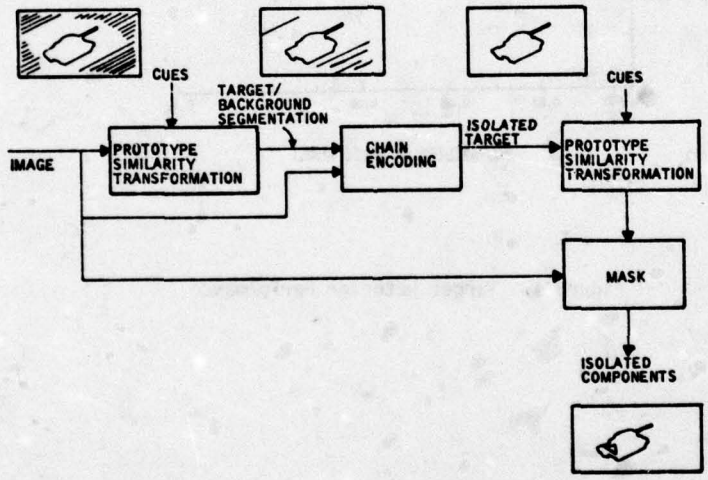


Figure 3. Component Extraction of Tactical Targets Through Iterative Use of Prototype Similarity.

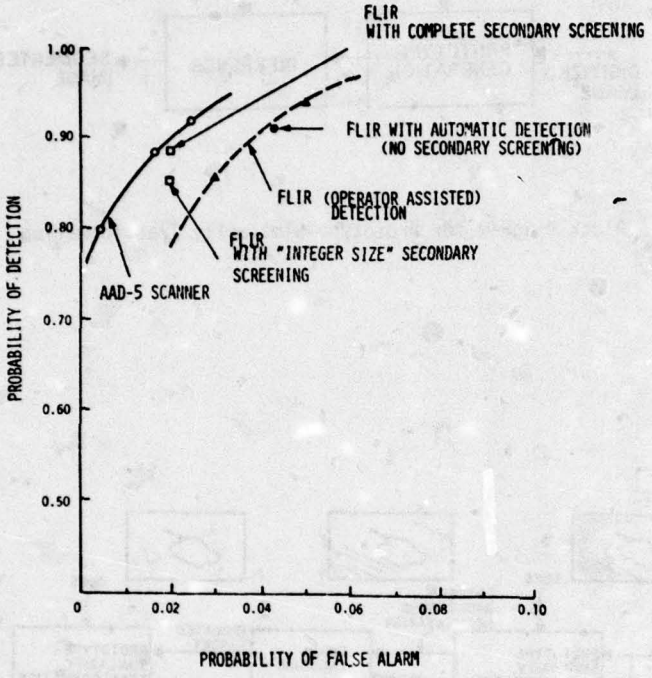


Figure 4. Target Detector Performance

IMPLEMENTATION OF ADVANCED REAL-TIME IMAGE UNDERSTANDING ALGORITHMS

by

G.R. Nudd, P.A. Nygaard,* S.D. Fouse, and T.A. Nussmeier

Hughes Research Laboratories
3011 Malibu Canyon Road
Malibu, CA 90265

Abstract - This paper describes the work undertaken during the past six months by Hughes Aircraft Company under a subcontract from the University of Southern California, Image Processing Institute, for the DARPA Image Understanding program. Two principal areas are discussed: (1) the work on a charge-coupled-device/metal-oxide-semcionductor test circuit to develop five real-time preprocessing operators and (2) the study and analysis of a real-time, high-level processor architecture that measures texture. Also, the design and development of a real-time processing facility based at Hughes Research Laboratories for performance test and evaluation is described.

I. INTRODUCTION

During the past six months, we continued our work to develop custom-designed integrated circuits for real-time implementation of image-understanding algorithms. The work has centered on three areas: the detailed design and layout of a third test chip, TCIII; the development of new concepts for more advanced (higher level) processing operations (including a texture chip);[1] and the design and construction of the necessary circuits, such as clock drivers, to operate the processors.

In the previous phase of this program, we developed concepts and test circuits for "real-time" (equivalent to television data rates) processing of "low-level," or preprocessing, algorithms, including edge detection, unsharp masking, local averaging, adaptive stretch, and binarization. Our approach is to employ fast analog preprocessors integrated at or close to the sensor itself and then to follow this by programmable digital processing using highly regular LSI or VLSI designs. In the preprocessing stage, which includes image-enhancement, feature analysis, etc., where a limited number of operations are required on each pixel, the effective data rates are extremely high (typically in excess of 10^8 operations/sec). This exceeds the current throughput of state-of-the-art high-density digital ICs (10^7 operations/sec). For this processing, a direct analog approach that maintains an accuracy equivalent to 6 bits can be provided directly on the sensor itself. At the higher level, where many operations are required on each data point, but where the number of data elements is reduced, a digital approach providing an increased accuracy and dynamic range is most appropriate.

Section II of this paper describes our continuing work on Test Chip III to develop high-speed preprocssing functions. Section II describes an approach to higher level processing (such as texture and segmentation), and Section IV discusses the work we have undertaken on our image-processing facility to enable us to operate our custom-built ICs in real time.

II. DESIGN AND FABRICATION OF TEST CHIP III

We have investigated five circuits for inclusion in our current test chip. These are a 3 x 3 Laplacian operator, a 7 x 7 kernel (which is currently being implemented as an edge detector but can be mask programmed to perform other operations, such as the binary checker boards or unsharp masking), a 5 x 5 programmable filter (which we intend to integrate with a commercial microprocessor), a 5 x 5 "cross-shaped" median, and a large bipolar convolutional array for 26 x 26 pixel convolutions.

For each of these, we have developed circuit concepts that will allow the data to be processed at real-time data rates. Circuit simulations that evaluate the accuracy and speed and hence the dynamic range have been completed for each circuit. The detailed designs and layouts of these operators have now been completed, and we anticipate having processed parts by July, which should allow the preliminary evaluation to be started before the end of this phase of the contract in September.

The technology used to implement the algorithms is n-channel, two-phase metal-oxide semiconductor (MOS) and charged-coupled device (CCD). The full chip size is approximately 225 mils x 225 mils, and conventional photolithography is used, resulting in a linewidth of ~5 μm. With this resolution and by using surface-channel technology, a clock rate of 7.5 MHz should be possible. A block schematic and a brief description of each circuit is given below.

*Mr. Nygaard is with the Hughes Carlsbad Research Center.

A. Laplacian Operator

The Laplacian operator[2] is a bipolar weighting scheme $\underline{A}$ operating on a 3 x 3 array of picture elements, given by

$$\underline{A} = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} , \qquad (1)$$

to produce the convolution output $\underline{A} * \underline{p}$, where $\underline{p}$ is the unprocessed image array. It is used for crispening and edge sharpening. It can be implemented directly at the sensor using a two-dimensional CCD array consisting of a set of linear transversal filters. A schematic of the system is shown in Figure 1. Each filter is a two-phase, n-channel device with 18 gates. The added latency time for this device is equivalent to ∿0.5 pixels (≈0.1 μsec). This is in addition to the inherent delay of the algorithm of approimxately one video line (∿63 μsec).

The circuit uses the floating gate technique to sense nine adjacent charge packets representing the array of 3 x 3 adjacent pixels. Three adjacent lines of charge, representing the video data, are clocked through the array, and the weighted sum of the charge or pixel magnitudes at each clock sample is applied to an "on-chip" sample and hold. The voltage on the floating gate array, sensed by the sample and hold at the nth clock period, T, is

$$V_o(nT) = \frac{\sum_{i=1}^{9} A_i Q_i(nT)}{C_A} , \qquad (2)$$

where C_A is the total capacitance of the array and connecting bus-line, $Q_i(nT)$ is the total charge representing each picture element under the gates at time nT, and A_i is the effective gate area at each location. For correct operation, the length of each gate must be proportional to the elements of $\underline{A}$. Since the length of each gate must be a positive value, the conventional approach would be to implement $\underline{A}$ as

$$|\underline{A}| = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad (3)$$

and to connect each of the gates to either a positive or negative bus line of a differential

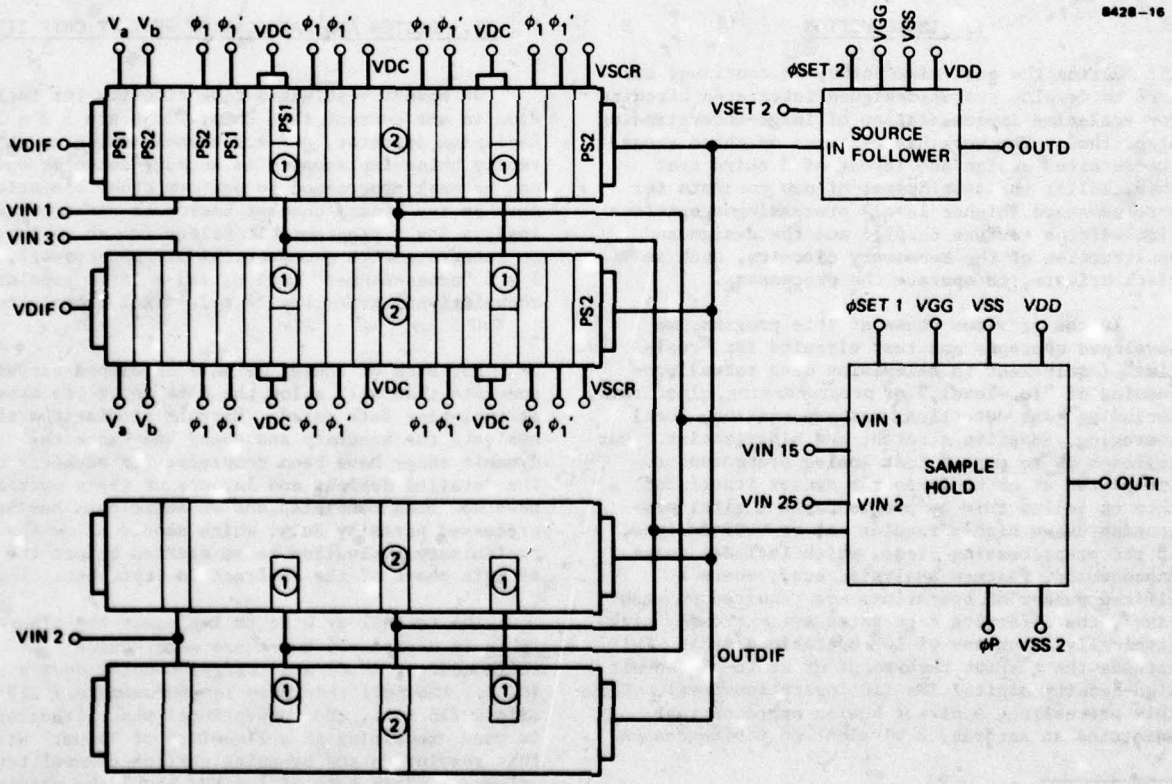


Figure 1. Schematic of CCD Laplacian operator.

amplifier. In practice, the differential amplifier, which itself can be implemented in n-MOS technology, becomes a significant portion of the total area of the chip, typically comparable or perhaps even larger than the CCD array itself. Further, the differential amplifiers are themselves voltage-controlled devices, and the charge-to-voltage transitions necessary can introduce linearities and noise. For this reason, we use the Hughes-patented displacement-charge-subtraction (DCS) technique,[3] which implements $\underline{A}$ directly as

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} \qquad (4)$$

and is low capacitance and eliminates "common-mode" output. This technique has been shown to provide up to 90-dB dynamic range and 68-dB common mode rejection.

The processed outputs from the array are fed directly to an "on-chip" sample and hold circuit, which eliminates clock feedthrough and rejects coherent noise sources. These circuits have been designed and simulated to run at a 7.5-MHz data rate with accuracy equivalent to 6 bits.

B. 7 x 7 Mask Programmable Kernel

In the April 1968 Semi-Annual Report, several processing algorithms were discussed that use a 7 x 7 array as a binary checkerboard weighting for image decomposition or as a version of unsharp masking or deblurring. Because of the interest in this array size, we have built a mask-programmable array that can be used to form a variety of operators. The basic concept consists of a 7 x 7 array of CCD stages that can be operated from seven parallel, adjacent video lines. The basic structure is shown in Figure 2. With this basic structure, we can use a mask change to perform a variety of different operations. Basically, only those levels that determine the filter weightings and the bus line interconnection need be changed to provide each of the operations discussed in the April 1978 Semi-Annual Report. This technique provides a very flexible and cost-effective way of performing a wide veriety of 7 x 7 algorithms.

Initially, we have designed the mask to perform a 7 x 7 edge-detection operation with radially symmetric weights. The weights used are given by

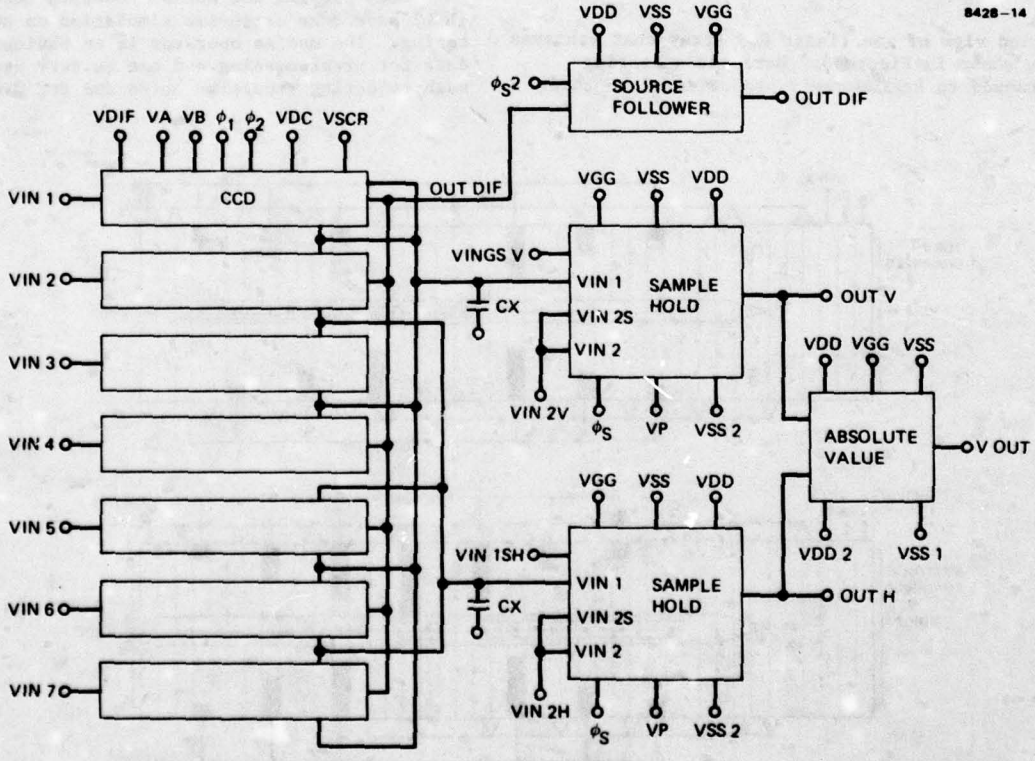


Figure 2. 7 x 7 Mask programmable array.

$$\underline{H}_x = \begin{bmatrix} 0.08 & 0.17 & 0.25 & 0.34 & 0.25 & 0.17 & 0.08 \\ 0.17 & 0.34 & 0.51 & 0.64 & 0.51 & 0.34 & 0.17 \\ 0.25 & 0.51 & 0.64 & 1.0 & 0.64 & 0.51 & 0.25 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.25 & -0.51 & -0.64 & -1.0 & -0.64 & -0.51 & -0.25 \\ -0.17 & -0.34 & -0.51 & -0.64 & -0.51 & -0.34 & -0.17 \\ -0.08 & -0.17 & -0.25 & -0.34 & -0.25 & -0.17 & -0.08 \end{bmatrix} \quad (5)$$

$$\underline{H}_y = \begin{bmatrix} 0.08 & 0.17 & 0.25 & 0 & -0.25 & -0.17 & -0.08 \\ 0.17 & 0.34 & 0.51 & 0 & -0.51 & -0.34 & -0.17 \\ 0.25 & 0.51 & 0.64 & 0 & -0.64 & -0.51 & -0.25 \\ 0.34 & 0.64 & 1.0 & 0 & -1.0 & -0.64 & -0.34 \\ 0.25 & 0.51 & 0.64 & 0 & -0.64 & -0.51 & -0.25 \\ 0.17 & 0.34 & 0.51 & 0 & -0.51 & -0.34 & -0.17 \\ 0.08 & 0.17 & 0.25 & 0 & -0.25 & -0.17 & -0.08 \end{bmatrix}$$

and the edge-detected output can be written as

$$S_i = |\underline{P}*\underline{H}_x| + |P*\underline{H}_y| \quad . \quad (6)$$

A detailed view of one linear CCD array that achieves this is shown in Figure 3. Here the weightings are arranged to be inversely proportional to their distance from the center of the array. In this way, the edge value is concentrated ("focused") towards the center picture value, and the larger array size gives greater immunity to noise in the sensed image.

C. 5 x 5 Programmable Array

To achieve simultaneously the high-speed capability and the added flexibility of programmable operations, we have included a data-programmable 5 x 5 array as shown in Figure 4. The programmable approach should allow many of the image-understanding operations of interest on a 5 x 5 array to be performed with one circuit. The concept is shown in Figure 5. It has been designed to accept data at the standard 7.5-MHz video rate and to enable the weighting functions to be changed at the frame rate of 30 Hz. Since each weighting node has been brought out directly to an external pin of the 64-pin package, we can independently vary each element of the 25-point convolution with an accuracy of ∿2%. Further, since our aim is to drive the weights from a commercial microprocessor, we can in effect cancel out many of the processing inaccuracies and nonlinearities to obtain optimum performance, enabling us to study the analog-digital or low-level/high-level interfaces.

D. 5 x 5 "Plus-Shaped" Median Filtering

Both USC IPI and Hughes Research Laboratories (HRL) have done extensive simulation on median filtering. The median operator is an obvious candidate for preprocessing and can be very useful for both rejecting impulsive noise and for overcoming

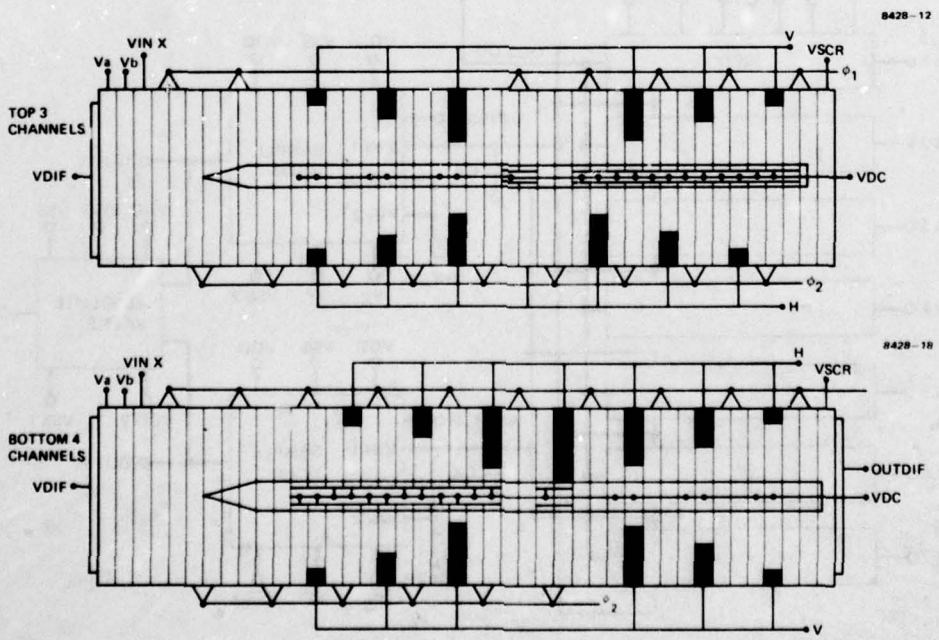


Figure 3. Mask structure programmed for 7 x 7 edge detection.

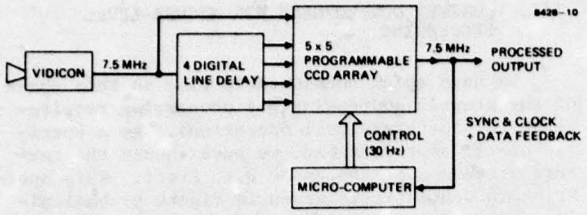


Figure 4. Schematic of test system for programmable filter incorporating mirco-computer control.

defects in the imaging system. Both HRL and IPI studies show that a "plus-shaped" array with 9 pixels is optimal for many of the images of interest. Perhaps the most direct approach to a median filter is to perform a sort operation and then choose the fifth element in the stack (for a 5 x 5 cross). To do this, $n(n-1)/2 = 36$ comparators are required. The conventional approach is to form the ladder network, or "bubble-sort" array, shown in Figure 6. Here each comparator module (CM) has three basic states, depending on the relative magnitude of the two inputs "a" and "b." In the configuration shown, if $a > b$ or $a = b$, the CM acts simply as two parallel, one-element delays. For $b > a$, however, it acts as a cross-bar switch and reverses the two outputs. The effect after 36 comparisons is to provide 9 parallel outputs ordered by increasing magnitude, with the center output being the median value.

This structure can be built directly in MOS/LSI using MOSFETs to provide a result equivalent to 7 bits at a rate of 7.5 MHz. It can also be built into a modular design, which will allow the array size to be increased by adding parallel chips. Our present design is a direct MOS implementation that uses external delays to determine the pixel array shape. In this way, the operation can be performed over a variety of kernels.

E. 26 x 26 Bi-Polar Convolution Filter

We have included on this chip a processing algorithm suggested by Professor David Marr and his colleagues at MIT. From a technology standpoint, it is interesting because it has a significantly larger kernel size than the arrays built to date and requires high accuracy. The full kernel consists of a 26 x 26 element array with a weighting ratio of approximately 1:150. We have used the circular symmetry of the array to build only a

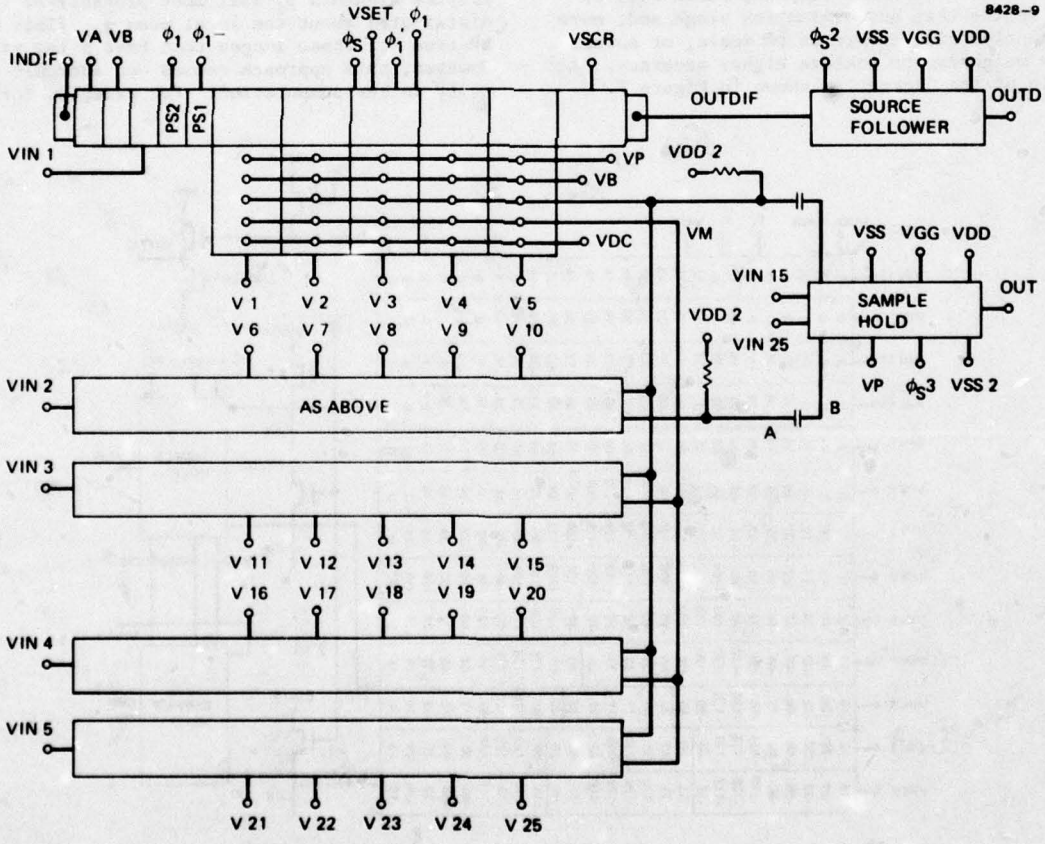


Figure 5. Schematic of 5 x 5 programmable filter.

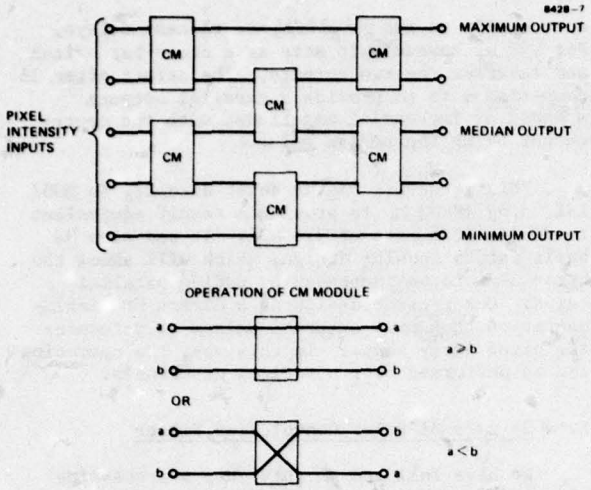


Figure 6. Schematic of real-time median filter operator for 5 pixels.

26 x 13 element kernel and intend to use two chips with modified input structures for the full convolution. Further, we have decided to build the array with three separate outputs, which will be helpful in the test and evaluation stage and, more significantly, will enable us to scale, or normalize, the weighting to achieve higher accuracy. A schematic of the circuit is shown in Figure 7.

III. CONCEPT DEVELOPMENT FOR HIGHER-LEVEL PROCESSING

We have spent considerable time in this phase of the program addressing the processing requirements for the high-level operations. As a specific operation to analyze, we have chosen the texture processor of Professor W.K. Pratt. This operation (a schematic is given in Figure 8) basically consists of a Sobel or Laplacian operation followed by moment operations of the form

$$M_n = \sum^{N} (p_i - \bar{p})^n \quad ,$$

where p_i is the analog picture intensity, $\bar{p}$ is the average picture intensity, N is the number of pixels in the kernel, and n is the order of the moment (typically 1 through 4).

Assuming an input image with 6-bit dynamic range, the required output dynamic range could under the worst condition be 24 bits.

At first sight, it might appear that if $(p_i - \bar{p})^n$ were calculated directly, $(p_i - \bar{p})$ would typically be a small number since the individual picture elements p_i will most probably be closely distributed about the local mean $\bar{p}$. (This will be true for those images that have a low variance.) However, this approach causes considerable difficulty in the computation. For example, for each

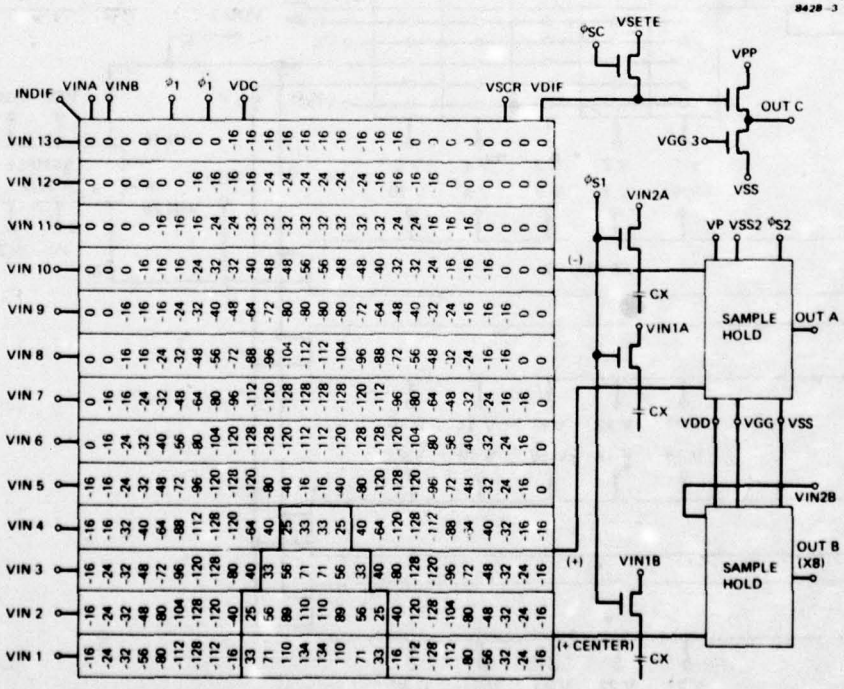


Figure 7. 26 x 13 pixel array implemented on test chip III.

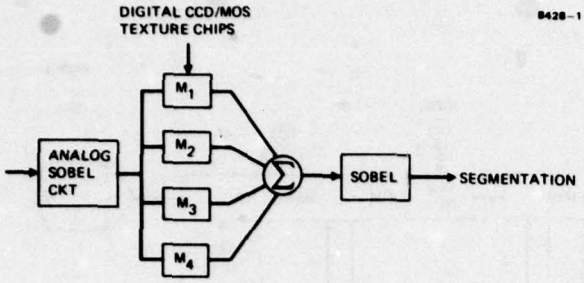


Figure 8. Concept of texture processing using moment calculations.

new picture element, which will occur approximately every 100 nsec, we will be required to calculate $\Sigma(p_i - \bar{p})^n$ in its entirety because $\bar{p}$ will also change at the pixel rate. For a 15 x 15 window, calculating just the first four moments, this will result in a throughput of 1.3×10^{10} operations per second, the majority of which will be multiplications. This is clearly an inappropriate approach since a high-speed multiply might take 50 nsec or more in the fastest high-speed emitter-coupled-logic technology. Several hundred channels, requiring a very large amount of power and circuity, would be required in a parallel architecture.

Clearly a preferable approach is to calculate the non-centered moments

$$M_1 = \frac{1}{N} (p_i)$$

$$M_2 = -\frac{1}{N^2} (p_i)^2 + \frac{1}{N} p_i^2$$

$$M_3 = -\frac{1}{N^3} (p_i)^3 - \frac{3}{N^2} p_i^2) + \frac{1}{N} p_i^3$$

$$M_4 = -\frac{3}{N^4} (p_i)^4 + \frac{6}{N^3} p_i^2 (p_i)^2$$

$$- \frac{4}{N^2} p_i^3 (p_i) + \frac{1}{N} p_i^4 .$$

In this way, the partial products can be calculated and stored, and with each new pixel we are required only to subtract the contributions of the oldest pixel from the summation and to add the newest. This can reduce the calculation rate by more than two orders of magnitude and enable real-time or near real-time operation.

IV. DEVELOPMENT OF TEST FACILITIES

We have now completed this design and begun fabrication of the facilities we will require to test and provide reference evaluation of this new chip. A schematic of this system is given in Figure 9.

REFERENCES

[1] O. D. Faugeras and W. K. Pratt "Stochastic-Based Visual Texture Feature Extraction," Image Processing Institute, Memorandum, USC (1979).

[2] W. K. Pratt, Digital Image Processing (John Wiley and Sons), New York, 1978).

[3] P. R. Prince, T. J. Mahony, J. A. Sakula, D. G. Maeding, and N. Cuk, "Displacement Charge Subtraction CCD Transversal Filter," presented at the 1978 International Conference on the Application of Charge-Coupled Devices, San Diego, Calif., 25-27 October 1978.

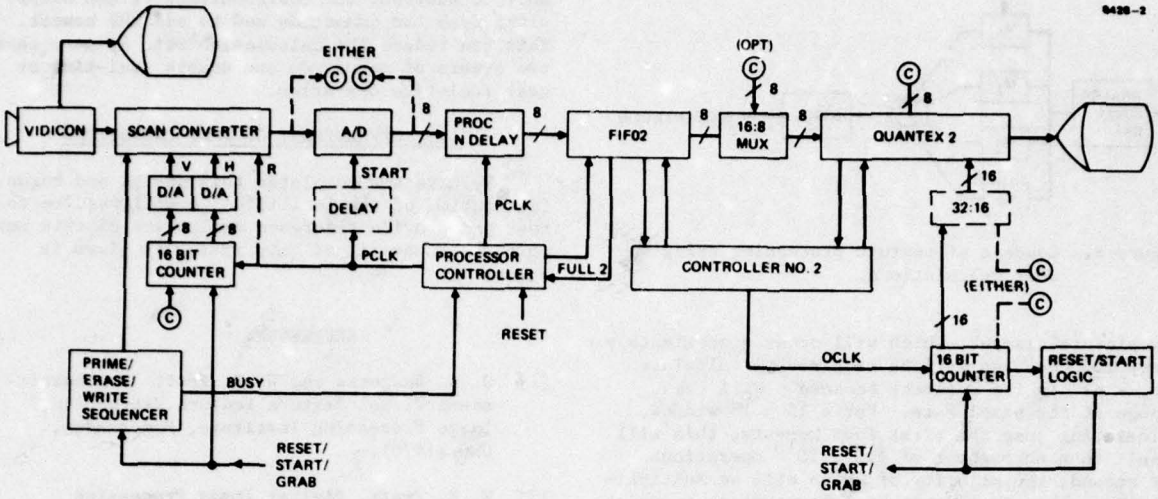


Figure 9. Schematic of test facility required for test chip III.

INVESTIGATION OF VLSI TECHNOLOGIES FOR IMAGE PROCESSING

W.L. EVERSOLE, D.J. MAYER, F.B. FRAZEE, and T.F. CHEEK, JR.

TEXAS INSTRUMENTS INCORPORATED
13500 NORTH CENTRAL EXPRESSWAY, P.O. BOX 225936
DALLAS, TEXAS 75265

ABSTRACT

This paper summarizes recent work performed for Carnegie-Mellon University on the investigation of very large scale integration (VLSI) implementations for image processing. Discussions of basic memory architectures needed for two dimensional operators, reconstruction of block truncation coded imagery data, and the implementation of a programmable sum of products operator are presented.

INTRODUCTION

The concept of a VLSI implementation of a digital image processor based on multiple real time arithmetic logic units (ALUs) and buffer memories was presented at the last workshop.[1] The implementation of the appropriate buffer memories and two image processing functions, reconstruction of block truncation coded data and programmable sum of products operator, are discussed below.

MEMORY ARCHITECTURES

In image processing systems having single line video inputs, memory is required to store previous pixel values for subsequent processing by two-dimensional operators such as edge detectors, median filter, etc. For our investigation a JxKxB bit two-dimensional input image and MxNxB bit two-dimensional operators are assumed.

For non-interlaced single-line video, the memory architecture of Figure 1 provides the necessary serial-to-parallel conversion needed for two-dimensional operators. The memory consists of (M-1) K-stage shift registers which format the data as M parallel input lines to the two-dimensional processors. The first line of the data need not be buffered. The buffer memory may be integrated onto the same IC as the processor to eliminate the need for M by B input pins on the processor IC. This integration would reduce the number of processor functions that could be integrated on a single IC, however it may be advantageous to duplicate the buffer memory on several different processors to reduce the total pin count and thus the size of the system.

For interlaced single-line video, a frame buffer memory is required to format the data into non-interlaced video. Two memory architectures for interlaced video inputs are shown in Figure 2. In Figure 2(a) a single-line-in single-line-out buffer

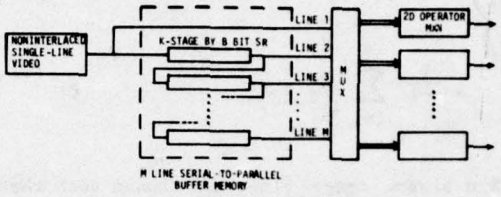
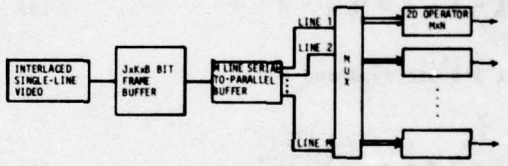
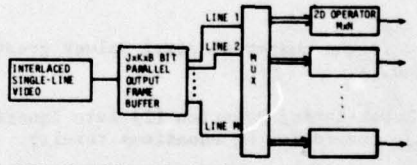


Figure 1. Memory Architecture for Non-Interlaced Single Line Video



a. Separate Frame Buffer and Serial-to-Parallel Converter



b. Combined Frame Buffer and Serial-to-Parallel Converter

Figure 2. Memory Architecture for Interlaced Single Line Video

inputs the video into the serial-to-parallel memory described in Figure 1. A single frame buffer memory may be used to store each video frame by using an alternating non-interlaced/interlaced addressing scheme to output one frame of reformatted data while reading interlaced data from the following frame.

In the memory architecture shown in Figure 2(b), both the frame buffer function and the serial-to-parallel conversion functions may be accomplished with a single frame store memory. This memory has a single line input and an M line parallel output. This approach requires a large pin-out (MxB).

BLOCK TRUNCATION CODING

Block truncation coding[2] techniques can be used to reduce the bandwidth needed to transmit imagery data. The sample mean and variance of small blocks of an image are used to statistically reconstruct the image from binarized image blocks. The following equations define the sample mean and variance, respectively.

$$\bar{X} = \frac{1}{N} \sum_{i=1}^{N} X_i \qquad (1)$$

$$\bar{\sigma}^2 = \frac{1}{N-1} \sum_{i=1}^{N} (X_i - \bar{X})^2 \qquad (2)$$

The binary image block is chosen such that all pixel values greater than $\bar{X}$ are set to 1 and all others are set to 0. In reconstruction all 0's are replaced by

$$A = \bar{X} - \bar{\sigma} \sqrt{\left(\frac{q}{N-q}\right)\left(\frac{N-1}{N}\right)} \qquad (3)$$

and all 1's are replaced by

$$B = \bar{X} + \bar{\sigma} \sqrt{\left(\frac{N-q}{q}\right)\left(\frac{N-1}{N}\right)} \qquad (4)$$

where q is the number of pixel values greater than the mean, $\bar{X}$.

Substituting Equation (2) into Equations (3) and (4), the following equations result:

$$A = \bar{X} - \sqrt{Y\left(\frac{q}{N-q}\right)} \qquad (5)$$

$$B = \bar{X} + \sqrt{Y\left(\frac{N-q}{q}\right)} \qquad (6)$$

where:

$$Y = \frac{1}{N} \sum_{i=1}^{N} (X_i - \bar{X})^2 \qquad (7)$$

A digital implementation of the block truncation encoding algorithm for 4 by 4 pixel blocks has been investigated and was discussed at the last workshop.

A simplified block diagram of the reconstruction circuit is shown in Figure 3. The reconstruction circuit receives the binary image block, counts the number of 1's (q) and uses a ROM look up procedure to calculate the square root quantity. A final adder/subtractor completes the calculation of A and B. The major portion of the circuit is the required ROM. The size of the ROM is determined byYand q. Since q can be any number from 0 to N andYmay be 8 bits, the ROM is (256xN) x 8 bits assuming an 8 bit answer is desired. For the case studied, N=16, therefore the ROM must have 4096 words. Furhter investigations have shown the largest value possible for the square root quantity in equations (5) and (6) is $\sqrt{255x15} \approx 62$, therefore only a 4096x6 bit ROM is necessary.

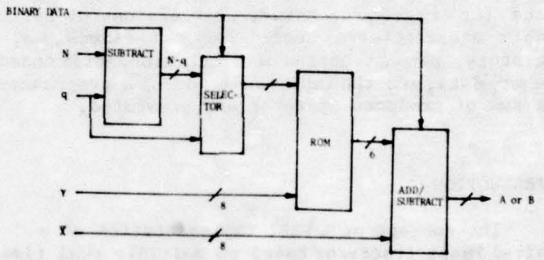


Figure 3. Implementation of Block Truncation Reconstruction

PROGRAMMABLE SUM OF PRODUCT OPERATOR

Many image processing algorithms require operations of the form

$$y = \sum_{\ell=0}^{L-1} a_\ell X_\ell \qquad (8)$$

where the a_ℓ's represent a set of fixed weighting coefficients and the X_ℓ's represent a set or a sequence of input values.

Equation (8) can be implemented using digital multipliers and adders; however, the size and power required to perform the multiplication at video data rates are prohibitive for most image processing applications. Much investigation has been performed recently on two similar techniques for the realization of Equation (8) with no digital multipliers.[3,4,5,6] The distributed arithmetic techniques implements the sliding sum of products (convolution) of an input word sequence with a set of weighting coefficients. The ROM-accumulator technique implements a nonsliding sum of products of an input word set with a set of weighting coefficients. Both methods use a table look-up procedure.

The operation performed by the distributed arithmetic technique is the convolution defined by

$$y_n = \sum_{\ell=0}^{L-1} a_\ell X_{n-\ell} \tag{9}$$

where $X_{n-\ell}$ is a B-bit word represented by

$$X_{n-\ell} = \sum_{j=0}^{B-1} X_{(n-\ell)_j} 2^j \qquad X_{(n-\ell)_j} \varepsilon \{0,1\} \tag{10}$$

Substituting Equation (10) into Equation (9) yields

$$y_n = \sum_{j=0}^{B-1} \left[\sum_{\ell=0}^{L-1} a_\ell X_{(n-\ell)_j} \right] 2^j \tag{11}$$

Since the values a_ℓ are fixed, the 2^L possible values of the bracketed term of Equation (11) may be calculated a priori and stored in a memory. For each j of the outer summation, the value of the bracketed term is recalled from the memory location whose address is formed by the L bit binary word

$$Z = [X_{(n)_j}, X_{(n-1)_j}, X_{(n-2)_j}, \ldots X_{(n-L+1)_j}] \tag{12}$$

The word stored in this location is given by

$$C(Z) = \sum_{\ell=0}^{L-1} a_\ell Z_\ell \qquad Z_\ell \varepsilon \{0,1\} \tag{13}$$

These values recalled from memory are weighted by the factor 2^j and summed over j.

Figure 4 shows a block diagram that implements Equation (9) using the table look-up algorithm of Equation (11). The sequential B_x-bit input signal, X, is loaded bit-serially into L-cascaded, B_x-bit delay lines. The last stage of each delay line is used to form the L-bit address for the memory. The shift and accumulate function performs the binary weighting and summation over j. The size of the memory needed for this implementation is 2^L words. If the weighting coefficients, a_ℓ, are of B_a-bit accuracy, the word size of the memory must be $B_a+\log_2 L$ bits to prevent overflow. The number, and thus the frequency, of table look-ups for this implementation is proportional to the input word length (B_x). The required memory can be reduced by partitioning the word sequence into Q blocks. This requires Q memories of size $2^{L/Q}$ words with each word $B_a+\log_2(L/Q)$ bits.

The penalty for blocking the memory is the addition of Q-1 digital adders. Figure 5 shows a Q=2 memory structure block diagram where L=4 words.

The frequency of table look-up can be reduced by multiple-bit addressing of the memory, i.e., by using M bits from each word to address the memory. The number of table look-ups is thus reduced to B_x/M. However, the memory size is increased to 2^{ML} words with each word $B_a+\log_2 L+ M$ bits. Figure 6 shows such a structure for which M=2. The last two stages of each delay line are used to form a 2L-bit memory address.

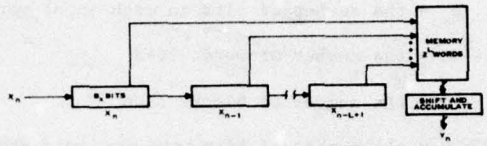


Figure 4. Block Diagram of Implementation of Single-Memory Storage of Partial Products

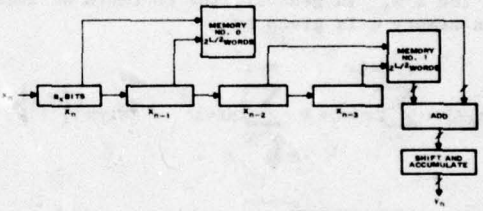


Figure 5. Block Diagram of Two-Memory Implementation

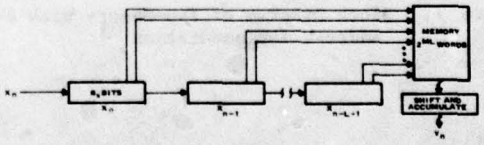


Figure 6. Block Diagram of Multiple-Bit Addressing Implementation

In general,

$$\text{Memory Size} = Q2^{ML/Q}\text{Words} \tag{14}$$

$$\text{Memory Word Size} = \begin{cases} B_a+\log_2(L/Q)\text{bits} & M=1 \\ B_a+\log_2(L/Q)+\ M\text{ bits} & M>1 \end{cases} \tag{15}$$

$$\text{Number of Digital Adders} = Q-1 \tag{16}$$

$$\text{Number of Table Look-Ups} = B_x/M \tag{17}$$

where: B_a = the number of bits in each weighting coefficient

B_x = the number of bits in each input word

L = the number of words used

Q = the number of blocks used

M = the number of bits from each word used for addressing.

A compromise among memory size, number of adders, and the number of table look-ups can be made to ease implementation. Figure 7 shows a block diagram implementation for B_x=6, L=4, Q=2, and M=2. The contents of the memories is a function of L, M, Q, and the a's. In general, the contents of location Z in memory q is given by

$$C(q,Z) = \sum_{\ell=0}^{L/Q-1} a_{qL/Q+\ell} \sum_{m=0}^{M-1} Z_{M\ell+m}2^m \qquad Z_{M\ell+m} \varepsilon \{0,1\} \tag{18}$$

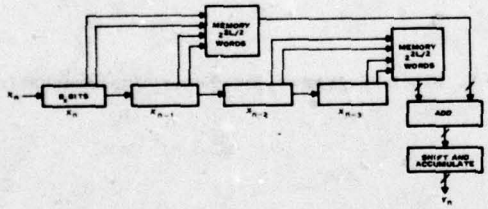


Figure 7. Block Diagram of Two-Memory With 2-Bit Address Implementation

The ROM-accumulator (RAC) technique implements Equation (8) exactly, i.e., without convolution. Representing each input word, X_ℓ, by

$$X_\ell = \sum_{j=0}^{B-1} X_{\ell_j} 2^j \qquad X_{\ell_j} \varepsilon \{0,1\} \tag{19}$$

and substituting into Equation (8) yields

$$y = \sum_{j=0}^{B-1} \left[\sum_{\ell=0}^{L-1} a_\ell X_{\ell_j} \right] 2^j \tag{20}$$

The RAC technique is a subset of the distributed arithmetic technique. Convolution of an input word sequence with the weighting coefficients, a, is not performed in the RAC technique. Figure 8 shows an implementation of Equation (8) using the RAC technique. The memory address is obtained from the bit-serial words, X, as shown. Equations (14) through (18) are still valid for the RAC technique. Like the distributed arithmetic implementation, the memory size and number of table look-ups can be reduced using memory blocking and multiple-bit addressing, respectively.

Several architectures and LSI technologies are being investigated as candidates for implementing a programmable sum of products operator. Many architectures have been deemed impractical because of the high internal operating frequency required. Investigation of other architectures which maximize processing time for the on-chip digital circuitry is continuing.

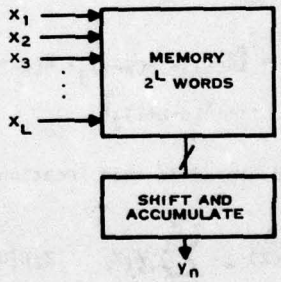


Figure 8. Implementation of Programmable Multiply and Sum Using ROM-Accumulator Technique

REAL TIME MEDIAN OPERATOR UNIT

The 5x5 median of medians[1] operator discussed in the last workshop is being implemented using off-the-shelf components. The breadboard is being designed for real time operation using a commerical TV camera as the sensor. The breadboard will allow evaluation of the median of medians operator and provide inputs for the design of the integrated version. A block diagram of the breadboard is shown in Figure 9. The memory needed to buffer 4 lines

of video will be partitioned to allow multi-resolution operation. The one line memory after the median operator is needed to refresh the display in low resolution operation. Operation will be performed on a single field of video to eliminate the need for a frame buffer memory.

CONCLUSIONS

This paper discussed the basic memory architectures needed for two dimensional operators, the reconstruction of block truncation coded imagery data and the implementation of a programmable sum of products operator. The design of a digital median operator breadboard for a 5x5 pixel window and 8 bit accurracy was described. Investigation of digital integrated circuits implementation of an IC capable of calculating the statistics of an image, i.e., the mean and variance or their analogs, the median and interquartile range, at real time video data rates has begun and the investigation of architectures for the programmable sum of products operator is continuing.

REFERENCES

1. W.L. Eversole, D.J. Mayer, F.B. Frazee, and T.F. Cheek, Jr., "Investigation of VLSI Technologies for Image Processing," Proc. IMAGE UNDERSTANDING WORKSHOP, Pittsburg, Penn. November 1978, pp. 191-195.

2. O R. Mitchell, E.J. Delp, and S.G. Carlton, "Block Truncation Coding: A New Approach to Image Compression," Conference Record, 1978 IEEE International Conference on Communications, pp 12B.1.1-12B.1.4.

3. C.S. Burrus, "Digital Filter Structures Described by Distributed Arithmetic," IEEE Transactions on Circuits and Systems, Vol. CAS-24, pp. 674-680, December 1977.

4. H.J. DeMan, C.J. Vandenbulcke, and M.M Van Cappellen, "High Speed NMOS Circuits for ROM-Accumulator and Multiplier Type Digital Filters," IEEE Journal Solid-State Circuits, Vol. SC-13, pp. 565-572, Octoer 1978.

5. T.A.C. Classen, W.F.G. Mecklenbrauker, and J.B.H. Peek, "Some Considerations on the Implementation of Digital Systems for Signal Processing," Phillips Research Reports, Vol. 30, pp. 73-84, 1975.

6. "Programmable Image Processing Element," Request for Proposal No. F33615-79-R-1763, Aeronautical Systems Division/PMREA Wright-Patterson AFB, Ohio, January 4, 1979.

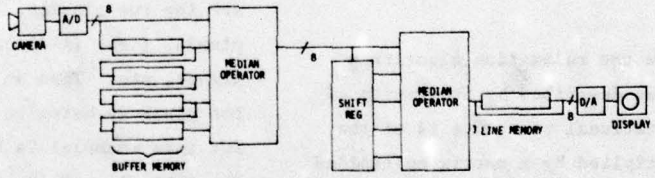


Figure 9. Block Diagram of 5 x 5 Median Operator Breadboard

RELAXATION, SYSTOLIC ARRAYS AND UNIVERSAL ARRAYS

Thomas J. Willett
Charles W. Brooks
Glenn E. Tisdale

Westinghouse Systems Development Division, Baltimore, Maryland 21203

ABSTRACT

Under contract to University of Maryland, Westinghouse has been implementing algorithms for the image understanding process. The program is sponsored by DARPA and monitored by the Army's Night Vision and Electro-Optical Laboratory. Our objective is the examination of the latest bit-sliced technology and the design of innovative architectures that are highly parallel, high-speed, fault tolerant, and require both a small instruction set and a small volume. A key consideration is the relaxation process, which has been under intensive investigation at Maryland, and which offers the prospect of improved decisionmaking, at the expense of computational load.

This paper describes an architecture for implementing relaxation in near real time and a hardware family of arrays to facilitate the architecture.

INTRODUCTION

We first examine the relaxation algorithms, at the pixel level, as described by University of Maryland. The computational structure is of the form of a scalar multiplied by a matrix multiplied by a vector. Implementation in a Systolic* Array architecture is described and hardware counterparts are found in Universal Arrays.** Because relaxation represents the new generation of higher order algorithms, Universal Array implementation is appropriate to produce speeds necessary to show feasibility.

*"Systolic Array" is a term used by H.T. Kung to describe a network of processors which rhythmically compute and pass data through the system. See Reference 3.

**"Universal Array" is the name given by Westinghouse to a functional array. See Reference 4.

RELAXATION AT PIXEL LEVEL

Relaxation at the pixel level, as developed thus far by the University of Maryland[1,2], is concerned, among other things, with curve detection and detection of light and dark regions in an image. This algorithm classifies each pixel into one of several classes by initially assigning to each pixel a probability of possible class membership based on local properties. The relaxation process then uses information about label (class) interactions on a local level to improve this prior classification.

For the light-dark classification problem, figure 1 shows a flow chart of the necessary computations which are now described. Suppose $\lambda\lambda'$ are the two classes, light and dark, for adjacent pixels, i.e., $(\lambda\lambda')$ can be (light, dark), (light, light), etc. Then an initial probability estimate for pixel xy being in class λ (assume λ = dark for this example) is $P_{xy}(\lambda) = (gl_{xy} - gl_{min})/gl_r$. The term gl_{xy} is the gray level at pixel xy, the term gl_{min} is the minimum gray level over the entire image, and gl_r is the range of gray levels over the image. An estimate of the probability of any pixel in the image of pixels having label λ is $\bar{p}(\lambda) = \frac{1}{n} \sum_{xy} p_{xy}(\lambda)$. An estimate of the joint probability of a pair of adjacent pixels, xy and x+i, y+j, having labels λ and λ' is $p_{ij}(\lambda\lambda') = \frac{1}{n} \sum p_{xy}(\lambda)\, p_{x+i,y+j}(\lambda')$. Computationally, xy is any pixel position in the image; x+i, y+j stands for each of the eight neighbors of the center pixel x,y in a 3×3 window as shown in figure 2. The local information about class interaction is being developed here. As the 3×3 window moves over the image, eight (8) different $p_{ij}(\lambda\lambda')$ expressions

are developed and cumulated, one for each of the neighbor positions. Further, there are four (4) possible ($\lambda\lambda'$) combinations, so there are actually 32 $p_{ij}(\lambda\lambda')$ expressions developed. Finally, the compatibility coefficients $r_{ij}(\lambda\lambda')$ expressions are developed. The compatibility coefficients $r_{ij}(\lambda\lambda') = p_{ij}(\lambda\lambda')/\overline{p}\ (\lambda)\ \overline{p}(\lambda')$ can be calculated from these expressions. These expressions $p_{ij}(\lambda\lambda')$, $\overline{p}\ (\lambda')$, $\overline{p}(\lambda)$, and $r_{ij}(\lambda\lambda')$ are referred to in figure 1 as properties of the image frame and must be obtained before the relaxation iterations can be started, which are now described.

The rationale for the various expressions in the relaxation iteration was described earlier.[3] To update the $p_{xy}(\lambda)$ expression based on local properties, eight (8) intermediate expressions $q_{ij}^{k+1}(\lambda) = \sum_{\lambda'} p_i^k(\lambda)\ p_j^k(\lambda')r_{ij}(\lambda\lambda')$ are calculated for each neighbor of the center pixel of the 3×3 window location. The superscript k refers to the iteration. Further, $p_{ij}^{k+1}(\lambda) = q_{ij}^{k+1}(\lambda)/\sum_{\lambda'} q_{ij}^{k+1}(\lambda')$, and $p_i^{k+1}(\lambda) = \frac{1}{n'}\sum_j p_{ij}^{k+1}(\lambda)$ are computed successively. Figure 3 shows each of the expressions expanded for two (2) classes, 1 and 2. Note that $p_{ij}^{k+1}(\lambda)$ is the updated version of $p_{xy}(\lambda)$, where n' = 8 and corresponds to each of the eight neighbors of a 3×3 neighborhood. These expressions $q_{ij}^{k+1}(\lambda)$, $q_{ij}^{k+1}(\lambda')$, $p_{ij}^{k+1}(\lambda)$ and $p_i^{k+1}(\lambda)$ are referred to in figure 1 as the iteration cycle. Consider now the hardware implementation of image frame properties and relaxation expressions in reverse order.

IMPLEMENTATION WITH SYSTOLIC ARRAY ARCHITECTURE

Hardware implementation can be examined as an array processor associated with the mathematical computation structure or associated with the geometry of a 3×3 window. For the time being the former approach is used. The expressions $q_{ij}^{k+1}(\lambda)$ and $q_{ij}^{k+1}(\lambda')$ may be expressed as matrix multiplications as shown in figure 4. But the complete expression, shown in figure 5, where the $\begin{pmatrix} p_i^k(\lambda) \\ p_i^k(\lambda') \end{pmatrix}$ term is really a pair of scalars in which the $p_i^k(\lambda)$ term is a scalar for the first row and $p_i^k(\lambda')$ is a scalar for the second row. So the first part of the relaxation iteration structure has been reduced to the case of a scalar multiplied by a matrix multiplied by a vector. An appropriate processor array architecture for this form is the Systolic Array.[3]

Consider a linear array of processors, each capable of communicating with the other and each of which has three input ports and two output ports. Further, each is capable of a multiply, an add, and holding three pieces of data in internal registers as shown in figure 6 .

The $p_j^k(1)$, $p_j^k(2)$ terms enters from the left and move to the right through the array; the $q_{ij}^{k+1}(\lambda)$, $q_{ij}^{k+1}(\lambda')$ terms move from right to left through the array and the matrix $r_{ij}(\lambda\lambda')$ terms move from the top to the bottom of the array and are staggered in time with regard with their entry. Then for a series of clock cycles, the data can be followed through the array as shown in figure 7. At clock cycle 1, $p_j^k(1)$ enters from the left and $q_{ij}^{k+1}(\lambda)$ enters from the right. At clock cycle 2, $r_{ij}(\lambda,1)$ enters the middle processor along with $p_j^k(1)$ and $q_{ij}^{k+1}(\lambda)$; $p_j^k(1)\cdot r_{ij}(\lambda,1)$ is formed and placed in $q_{ij}^{k+1}(\lambda)$. At clock cycle 3, $r_{ij}(\lambda,2)$ enters the left processor as does $p_j^k(2)$ and $q_{ij}^{k+1}(\lambda)$. Here $p_j^k(2)\cdot r_{ij}(\lambda,2)$ is formed, added to $q_{ij}^{k+1}(\lambda)$ resulting in $q_{ij}^{k+1}(\lambda) = r_{ij}(\lambda,1)\cdot p_j^k(1) + r_{ij}(\lambda,2)\cdot p_j^k(2)$. Similarly, in the right processor, $r_{ij}(\lambda',1)\cdot p_j^k(1)$ is formed and placed in $q_{ij}^{k+1}(\lambda')$. At the next clock cycle, $q_{ij}^{k+1}(\lambda)$ exits from the array and the two terms of $q_{ij}^{k+1}(\lambda')$ are formed in the center processor. Clock cycle 5, shows the matrix component $r_{ij}(\lambda',2)$ completely clear of the array with $q_{ij}^{k+1}(\lambda)$ and $p_j^k(1)$ also out of the array. Since the last step in forming $q_{ij}^{k+1}(\lambda)$ is a scalar multiplication by $p_i^k(\lambda)$, another processor, with the same internal structure, added to the array can accomplish this, as shown in figure 8. The array structure can be used to compute $\sum_{\lambda} q_{ij}^{k+1}(\lambda')$ since it is also of the form of a scalar

multiplied by a matrix multiplied by a vector.

Referring to figure 1, and the expressions for $p_{ij}^{k+1}(\lambda)$ and $p_i^k(\lambda)$, it is noted that the Systolic Array shown in figure 7 must be repeated for each of the eight neighbors of the center pixel yielding 32 processors, including scalar multiplication. Consider now the implementation of Systolic Arrays.

IMPLEMENTATION WITH UNIVERSAL ARRAYS

To avoid full frame storage it is necessary to complete the relaxation iterations as close to the frame used to compile the image statistics as possible. Secondly, as the classes expand, the required computations grow combinatorially. An arbitrary hardware goal at this point in the development is to complete relaxation in as close to real time as possible. With reference to the Systolic Array and the computation of the $q_{ij}^{k+1}(\lambda)$, $q_{ij}^{k+1}(\lambda')$ terms, this means completing three or four multiplications, three adds, and four shifts in approximately 100 nanoseconds for a typical 500×600 pixel frame size. The clock cycle governing the Systolic Array is then of the order of 10 nanoseconds. As the speed requirement becomes clearer and the small volume constraint is unchanged, the technology being developed for the next generation airborne signal processors becomes more significant and the Universal Arrays are now described.

Universal Arrays[4] or functional arrays are not to be confused with gate arrays, of which the field programmable logic array (FPLA) is an example, or the gate arrays currently used in main frame computers. The Universal Array consists of various transistor and resistor parts and is divided into functional cells rather than individual gates. The Westinghouse ECL Universal Array is a combination of diffusions and ion implantations which define a fixed set of transistors and resistors arranged in three types of cell groupings: internal logic cell, input/output cell, and input cell. Each type of cell is customized by configuring the two level metal interconnect pattern.

The array is composed of 48 internal cells in a 6×8 matrix, 24 input/output cells, and 16 input cells. The chip size is 228 mils by 240 mils, with power dissipation of 2.5 watts, and propagation delays of less than 6 nanoseconds. The chip is shown in figure 8, and figure 9 shows the internal logic cell. The array can be customized into a number of so-called "personalization." Under an Air Force contract, a number of personalizations will be produced and one is in probe test now.

This personalization is of great interest to us in implementing Systolic Arrays for relaxation because it is a 4×4 self-contained, multiply chip with an 8-bit product obtained in 12 nanoseconds. This is of the order of time needed to do relaxation in real time and 32 of these chips would fit in an area approximately 1-1/2 inches by 1-1/2 inches.

In future work, we shall be applying the Westinghouse Universal Array to relaxation and other University of Maryland algorithms in an attempt to approach real time processing capability and still stay in a small volume.

REFERENCES

1. Determining Compatibility Coefficients for Curve Enhancement Relaxation Processes, Peleg, S. and Rosenfeld, A., IEE Transc. on Systems, Man, and Cybernetics, Vol. SMC-8, No. 7, July 1978.

2. Work on Dark/Light Pixel Classification in progress by Univ. of Md. under DARPA/NVL sponsorship, but not published yet.

3. Hardware Implementation of Image Processing Using Overlays, Willett, T.J., DARPA Image Understanding Symposium, p. 175, November 1978.

4. Systolic Arrays for VLSI, Kung, H.T. and Leiserson, C.E., Dept. of Computer Science, Carnegie Mellon University, Dec. 1978.

5. LSI Universal Array Circuits for High Speed Programmable Processors, Natale, M.R. and Brooks, C.W., 1978 NAECON paper, Westinghouse Electric Corp., Systems Development Division, Baltimore, Md., March 1979.

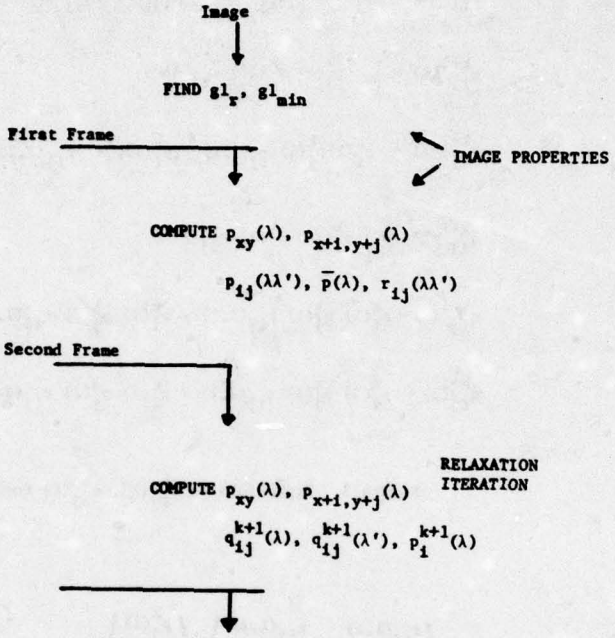


Figure 1. Flow Chart of Relaxation Computations

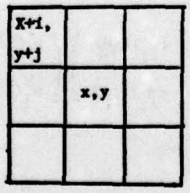


Figure 2. 3×3 Neighborhood

Let $\lambda = 1$, dark; $\lambda' = 1,2$

$$p_i^{k+1}(1) = \frac{1}{8} \sum_{j=1,2,\ldots 8} p_{ij}^{k+1}(1)$$

$$p_{ij}^{k+1}(1) = q_{ij}^{k+1}(1) / \sum_{\lambda'} q_{ij}^{k+1}(\lambda') = q_{ij}^{k+1}(1) / [q_{ij}^{k+1}(1) + q_{ij}^{k+1}(2)]$$

$$q_{ij}^{k+1}(\lambda) = \sum_{\lambda'} p_i^k(\lambda)\, p_j^k(\lambda')\, r_{ij}(\lambda\lambda')$$

$$q_{ij}^{k+1}(1) = p_i^k(1) p_j^k(1) r_{ij}(1,1) + p_i^k(1) p_j^k(2) r_{ij}(1,2) \qquad (\lambda = 1)$$

$$q_{ij}^{k+1}(\lambda') = \sum_{\lambda} p_i^k(\lambda')\, p_j^k(\lambda)\, r_{ij}(\lambda'\lambda)$$

$$q_{ij}^{k+1}(1) = p_i^k(1)\, p_j^k(1)\, r_{ij}(1,1) + p_i^k(1)\, p_j^k(2)\, r_{ij}(1,2) \qquad (\lambda' = 1)$$

$$q_{ij}^{k+1}(2) = p_i^k(2)\, p_j^k(1)\, r_{ij}(2,1) + p_i^k(2)\, p_j^k(2)\, r_{ij}(2,2) \qquad (\lambda' = 2)$$

Figure 3. Expansions of $p_{ij}^k(\lambda)$, $q_{ij}^k(\lambda)$ for Two Classes

$$\begin{pmatrix} r_{ij}(\lambda,1) & r_{ij}(\lambda,2) \\ r_{ij}(\lambda',1) & r_{ij}(\lambda',2) \end{pmatrix} \begin{pmatrix} p_j^k(1) \\ p_j^k(2) \end{pmatrix}$$

2×2 $\qquad$ 2×1

Figure 4. Matrix by a Vector Mulitplication

$$\begin{pmatrix} p_i^k(\lambda) \\ p_j^k(\lambda') \end{pmatrix} \begin{pmatrix} r_{ij}(\lambda,1) & r_{ij}(\lambda,2) \\ r_{ij}(\lambda',1) & r_{ij}(\lambda',2) \end{pmatrix} \begin{pmatrix} p_j^k(1) \\ p_j^k(2) \end{pmatrix} = \begin{pmatrix} q_{ij}^{k+1}(\lambda) \\ q_{ij}^{k+1}(\lambda') \end{pmatrix}$$

Scalar $\qquad$ 2×2 $\qquad$ 2×1

Figure 5. Multiplication Including Scalar

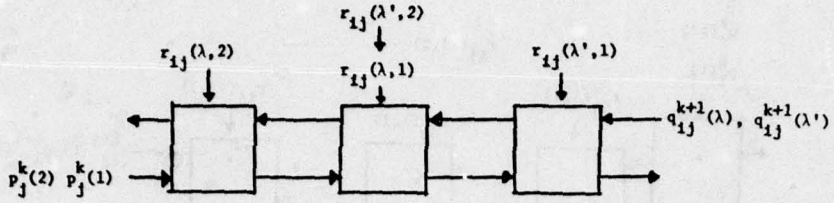


Figure 6. Systolic Array for Matrix-Vector Multiplication

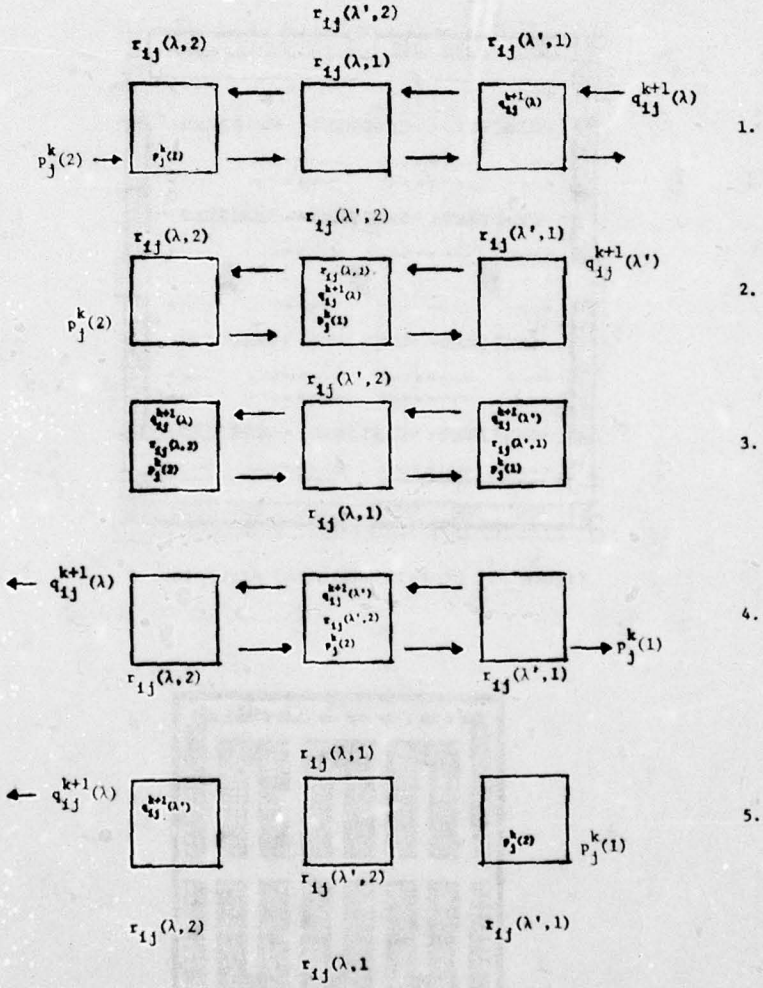


Figure 7. Processing Stages of Systolic Array

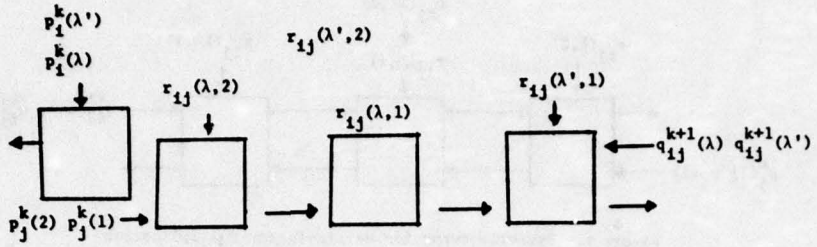


Figure 8. Complete Systolic Array

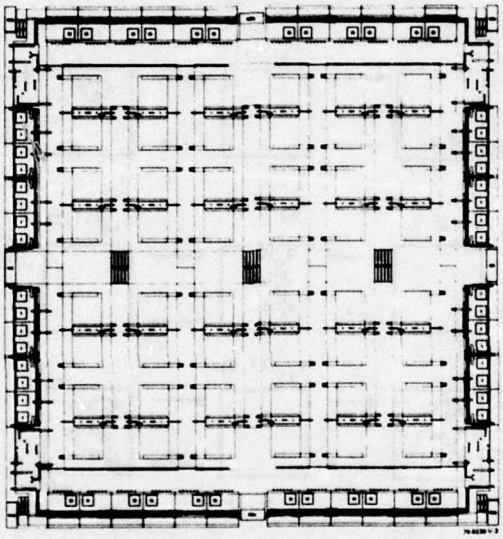


Figure 9. Overall Universal Array Layout

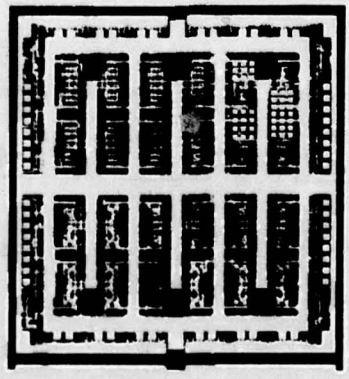


Figure 10. Photomicrograph of Personalized UA

SPARC--SYMBOLIC PROCESSING ALGORITHM RESEARCH COMPUTER

Gale R. Allen
Peter G. Juetten

Control Data Corporation
2800 East Old Shakopee Road
Minneapolis, Minnesota 55420

ABSTRACT

The SPARC program will result in a breadboard model of a high performance processor for image processing applications. This research project in computer architecture is being jointly performed by Control Data Corporation and Carnegie-Mellon University. The project is currently in the final design phase, with fabrication beginning. Results of trial processor coding are encouraging.

INTRODUCTION

The SPARC program is a research effort in computer architecture. Begun by ARPA in 1977, the program involves the design and fabrication of a breadboard model of a high-performance digital processor for use in image processing research. SPARC is a flexible, highly parallel computer, which may be used singly to offload computational tasks from a host general-purpose machine. The design also includes features which facilitate communication between multiple SPARC processors in array configurations for the large number of applications where processing power exceeding the capability of a single machine is required. The work is being performed as a joint effort between Control Data Corporation and Carnegie-Mellon University, with CDC responsible for hardware development and basic system software, while CMU is concerned primarily with user system software aids. CDC is concurrently supporting the construction of a second processor, and more extensive software development.

The goal of the present phase of the program is to have SPARC processors installed and operating at both CDC and CMU in the fall of 1979. At the present time, the hardware design is in its final stages. The electronic components are on order. Circuit board layout is in progress. Cabinetry is being assembled. Software work is in progress, with the main efforts being concentrated on the development of a SPARC cross-assembler, a register-level simulator, a basic operating system, and diagnostic programs. Trial coding of several applications algorithms has been performed. Work also continues on multiple processor array architecture and facilities.

SPARC ARCHITECTURE

A brief review of the SPARC processor architecture will facilitate understanding of the application example to be discussed later.

The basic processor, shown in block diagram form in Figure 1, consists of a number of hardware blocks, called functional units, each of which is designed to perform a certain set of operations on input data. The initial SPARC machine will contain six different unit types, plus a control unit to supervise and coordinate the operation of the processor. These units perform such operations as integer addition, multiplication, shifting, bit-by-bit logical functions, data storage, and input/output. The units are interconnected by a generalized crosspoint switching network, from which all units receive input operands and to which results are delivered. This crossbar switch, which is reconfigurable at a machine cycle rate, provides an internal data transfer capability of approximately 1×10^{10} bits per second.

Both the control and data interfaces between the functional units and the remainder of the processor are completely generalized, thus allowing the addition or substitution of different units as required to perform various specialized tasks. Units known as ring ports, designed for interprocessor communications in multiprocessor ring arrays, and memory access units, which provide links to high-capacity system memories, are currently in design. Other units proposed for implementation include an FFT unit, which would implement the butterfly type operation of the transform, and a floating point unit, which would provide hardware floating point arithmetic capability.

APPLICATION CODING RESULTS

The kernels of several algorithms from image processing application have been coded for a SPARC type processor in order to obtain estimates of processor performance. These include a linear interpolation process, called Warp Interpolation, in which a value is calculated for a hypothetical image pixel utilizing the magnitude and location of four adjacent pixels, a Dot Product algorithm,

which forms sums of various cross products between elements of two images in order to produce measures of correlation, a photonormalization process, and an FFT butterly using 32-bit complex arithmetic. These algorithms were chosen, in part, because code for them exists for the Flexible Processor, a current CDC machine which has been used as the basis of several image processing systems, thus enabling a direct performance comparison to be made.

Table 1 summarizes the performance comparisons between the two machines. A SPARC type machine with array processing features, such as ring ports and memory access units, was used, in order to obtain better correlation with the FP code. The results are encouraging, showing performance increase factors ranging between 17 and 59, coupled with a slight decrease in average coding time.

ARRAY PROCESSING ENHANCEMENTS

Mention has been made in this report of functional units under development at CDC which are intended to provide the communications facilities necessary to use multiple SPARC processors in array configurations, for applications requiring compute power measured in billions of operations per second. One such unit is the Ring Port, which serves as an interface between the processor and a ring communications system, which has been described in reference 1.

A second type of unit is intended to provide high bandwidth access for processors to megabyte size common system memories. This unit will feature 800 M-bit bandwidth, maximum addressing capability of 2^{35} bytes, and multiple modes of operation, including stream modes, in which fetch operations and address updating are controlled by the memory access unit hardware, and performed at a rate sufficient to keep the processor supplied with data. These units not only provide processor access to large amounts of memory, but serve as an additional communication device for multiple processors sharing common external storage.

STATUS

The electronic parts for the prototype SPARC computers are on order. Delivery is scheduled for early summer. Qualification lots of the three new LSI arrays designed for SPARC are in fabrication. Final design clean-up and simulation is in progress, and circuit board layout has begun. The cabinetry, including power supplies and refrigeration hardware, is being assembled.

In the software area, overall objectives for the initial versions of simulator, cross-assembler, diagnostics and operating system have been determined. Coding effort is currently taking place in these areas. Work has been initiated at CDC toward the definition and/or selection of a potential high-level language for the processor.

CONCLUSION

The SPARC program, jointly performed by CDC and CMU, is progressing toward the goal of developing a high-performance, modular, digital processor architecture with application to image processing problems requiring computational capability in the 10^8 - 10^{10} operations per second range. The architecture employed enables systems of one or many processors to be easily configured (and reconfigured) to meet various processing requirements. Facilities are provided to accommodate system components, such as memory devices, which are required to support the computing power of multiprocessor arrays.

The authors wish to acknowledge the contributions of many engineering, software, and management personnel at CDC, along with the significant contributions of the staff at CMU. The combined efforts of all concerned continue to be of great benefit to the project.

REFERENCES

(1) Gale R. Allen, Peter G. Juetten, "SPARC-SYMBOLIC PROCESSING RESEARCH COMPUTER". Proceedings: Image Understanding Workshop, Pittsburgh, PA, November, 1978, pg. 182-190.

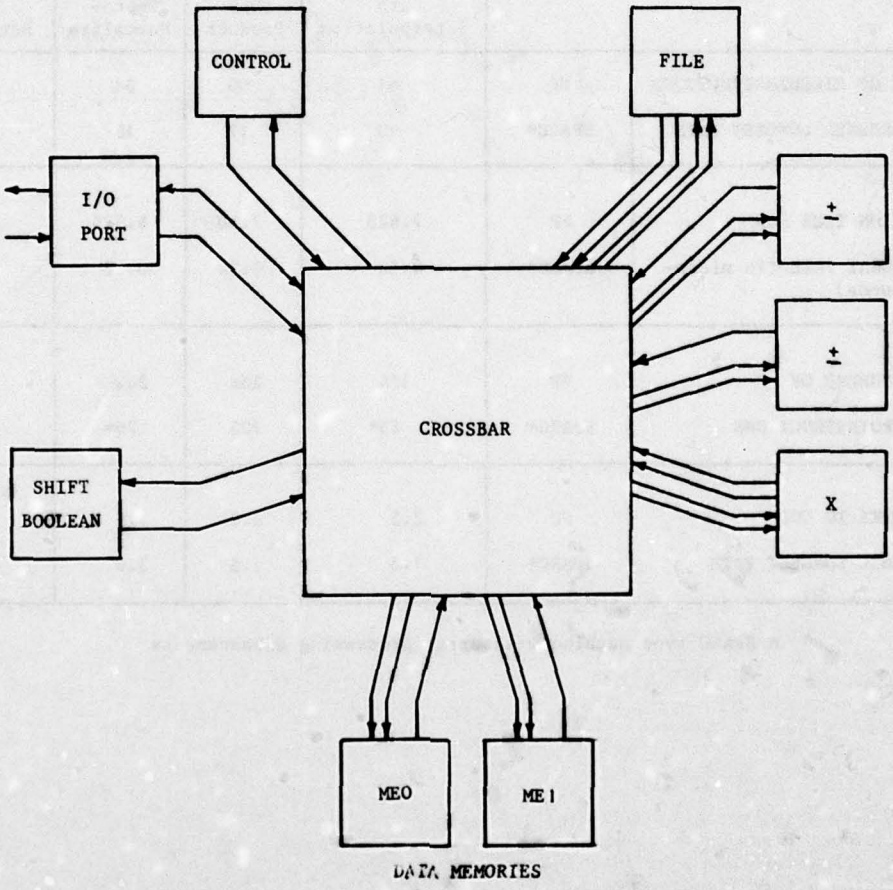


Figure 1. Basic SPARC Architecture

TABLE I. SOME SPARC*/AFP COMPARISONS

		ALGORITHM			
		Warp Interpolation	Dot Product	Photo-Normalize	FFT Butterfly
NUMBER OF MICROINSTRUCTIONS IN KERNEL LONGEST PATH	FP	61	60	51	172
	SPARC*	18	17	18	18
EXECUTION TIME FOR LONGEST PATH (in micro-seconds)	FP	7.625	7.50	6.375	21.5
	SPARC*	0.36	0.34	0.36	0.36
TOTAL NUMBER OF MICROINSTRUCTIONS	FP	184	206	204	N/A
	SPARC*	85*	102	70*	
MAN-WEEKS TO CODE KERNEL LONGEST PATH	FP	2.5	2.0	1.5	2.5
	SPARC*	1.5	1.5	3.0	0.4

* A SPARC type machine with array processing enhancements