TEC-0107

# Research in the Automated Analysis of Remotely Sensed Imagery

David M. McKeown, Jr., et al.

Carnegie Mellon University
School of Computer Science
Pittsburgh, PA 15213-3890

March 1998

Prepared for:
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington, VA 22203-1714

Monitored by:
U.S. Army Corps of Engineers
Topographic Engineering Center
7701 Telegraph Road
Alexandria, VA 22315-3864

US Army Corps of Engineers
Topographic Engineering Center

T
E
C

U.S. ARMY TOPOGRAPHIC ENGINEERING CENTER
TEC

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*

| | | |
|---|---|---|
| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE<br>March 1998 | 3. REPORT TYPE AND DATES COVERED<br>Second Annual Sep. 1993 - Sep. 1994 |

| | |
|---|---|
| 4. TITLE AND SUBTITLE<br>Research in the Automated Analysis of Remotely Sensed Imagery | 5. FUNDING NUMBERS<br>DACA76-92-C-0036 |
| 6. AUTHOR(S)<br>David M. McKeown, Jr. Steven D. Cochran Stephen J. Ford Stephen J. Gifford Wilson A. Harvey J. Chris McGlone Yuan Hsieh Jeffrey A. Shufelt Michael F. Polis Michel Roux | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Carnegie Mellon University<br>School of Computer Science<br>Pittsburgh, PA 15213-3890 | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>Defense Advanced Research Projects Agency<br>3701 North Fairfax Drive, Arlington, VA 22203-1714<br><br>U.S. Army Topographic Engineering Center<br>7701 Telegraph Road, Alexandria, VA 22315-3864 | 19. SPONSORING / MONITORING AGENCY REPORT NUMBER<br>TEC-0107 |
| 11. SUPPLEMENTARY NOTES | |
| 12a. DISTRIBUTION / AVAILABILITY STATEMENT<br>Approved for public release; distribution is unlimited. | 12b. DISTRIBUTION CODE |

13. ABSTRACT *(Maximum 200 words)*

This report presents an overview of Carnegie Mellon University's (CMU) program of research at the Digital Mapping Laboratory in the general area of the automated analysis of remotely sensed imagery. We report progress in the areas of digital photogrammetry, automatic building extraction, stereo analysis, multispectral image analysis, virtual world construction, and knowledge-based systems for scene interpretation.

| | |
|---|---|
| 14. SUBJECT TERMS<br>Site modeling Feature Extraction Information Fusion Database Intensification | 15. NUMBER OF PAGES<br>60 |
| | 16. PRICE CODE |

| | | | |
|---|---|---|---|
| 17. SECURITY CLASSIFICATION OF REPORT<br>UNCLASSIFIED | 18. SECURITY CLASSIFICATION OF THIS PAGE<br>UNCLASSIFIED | 19. SECURITY CLASSIFICATION OF ABSTRACT<br>UNCLASSIFIED | 20. LIMITATION OF ABSTRACT<br>UNLIMITED |

## TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

## LIST OF FIGURES (Continued)

## LIST OF FIGURES (Continued)

## LIST OF TABLES

## PREFACE

This research was sponsored by the Defense Advanced Research Projects Agency (DARPA) and monitored by the U.S. Army Topographic Engineering Center (TEC) under contract DACA76-92-C-0036, titled, "Research in the Automated Analysis of Remotely Sensed Imagery." The DARPA Program Manager was Mr. Oscar Firschein, and the TEC Contracting Officer's Representative was Ms. Lauretta Williams.

# RESEARCH IN THE AUTOMATED ANALYSIS OF REMOTELY SENSED IMAGERY: 1993–1994

David M. McKeown, Jr. Steven Douglas Cochran Stephen J. Ford
Stephen J. Gifford Wilson A. Harvey Yuan C. Hsieh
J. Chris McGlone Michael F. Polis Michel Roux
Jefferey A. Shufelt

Digital Mapping Laboratory
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890

## 1 Introduction

The automated compilation of man-made and natural terrain in urban or *built-up* areas has been the focus of our research for a number of years. Built-up areas provide some of the most difficult and time consuming tasks for the cartographic community, and provide a rich environment of varied structures and natural terrain features to test the robustness of new approaches to computer vision. The theme of our research in the Digital Mapping Laboratory is to understand the computational aspects of automated recovery of 3-D scene information using a variety of image domain cues. These cues include the analysis of cast shadows, stereo matching, geometric models, and structural descriptions based upon analysis and combination of low-level image-based features. We look for opportunities to augment traditional computational vision techniques with domain knowledge, since it is clearly used by both cartographers and imagery analysts in a variety of tasks ranging from mapping, to environmental land use analysis, to natural resource inventory. In this report we describe a variety of research activities jointly sponsored by ARPA, the U.S. Army Topographic Engineering Center (TEC), and the Air Force Office of Scientific Research (AFOSR).

Section 2 describes recent work in the application of rigorous digital photogrammetric methods to image orientation, projective geometry, and sensor modeling within the context of the ARPA RADIUS program. This research forms the solid geometric foundation for a variety of other feature extraction activities. In particular, our emphasis on object space analysis for monocular, stereo, and multiple image matching has been enabled by results in this area of research.

Section 3 details recent extension of our previous work in the generation of 2-D image space building boundaries using a single (monocular) nadir or oblique view of the scene. We give a detailed description of how these techniques have been extended to produce 3-D object space wire-frame buildings, measured in metric units.

Section 4 describes our current work on three stereo-related projects. The first is directed toward understanding the strengths and weaknesses of our existing stereo matching systems and to provide remedies to improve overall performance. We have integrated object-space knowledge into the stereo process to adaptively adjust the stereo search matching region. The second part explores a new area-based stereo approach that is based in object-space using multiple (more than two) images. The third examines object space feature matching for building delineation across multiple-images.

Section 5 describes recent work in generating high confidence surface material classmaps for use in multispectral fusion. The availability of moderate resolution (3–5 meter ground sample distance) multispectral imagery provides an opportunity to use image understanding techniques in addition to statistical analysis provided by traditional remote sensing methods. The production of detailed surface material maps has application in spatial database construction for simulation, land use analysis, and as an additional source of information in cartographic feature extraction.

Section 6 details our recent work on improving the scalability of large knowledge-based systems for image analysis. We report on recent work using a portable distributed shared memory system (MIDWAY) to support our previous work in task-level parallelism. Additionally, we present investigations in the use of alternate computational models to support explicit parallelism for analysis systems that use large amounts of image data combined with spatial and structural knowledge.

Finally, Section 7 gives an overview of our work in the domain of virtual world database generation for advanced distributed simulation. This includes our efforts to generate spatial databases from standard sources such as DMA ITD and DTED. We describe the problems encountered in combining these data sources as well as several experiments toward the integration of features extracted automatically from aerial imagery.

## 2 Integration of Photogrammetry and Computer Vision

A standard assumption in computer vision research is that no external information is available about the image, the sensor, or the scene; the only usable information is what can be inferred from the image itself. When working in a cartographic application domain, such an assumption is unnecessarily limiting and essentially unrealistic. Images from sensors with unknown internal geometry, or taken from a completely unknown viewpoint are seldom, if ever, used in a production cartographic environment. Approximate sensor position and orientation is often available from platform navigation sensors, such as global positioning systems (GPS), or from orbital information, and can be refined using control point or geometric scene information.

By definition, cartography is directly concerned with the world. The production of cartographic information must therefore be done within a context of world knowledge, including both general knowledge of typical scene geometry and specific knowledge about the particular scene of interest. Coupling generic knowledge, such as typical forms of man-made structures with specific scene knowledge such as terrain models, digital elevation models (DEMs), or digital map data (DMA ITD), allows the design of effective, efficient algorithms to extract man-made cartographic features and place them within a rigorous metric framework.

The integration of sensor and scene knowledge into vision algorithms requires a rigorous photogrammetric framework describing image and scene geometry, closely tied to existing cartographic databases. Work done in the past few years at the Digital Mapping Laboratory has concentrated on this integration; first, on building a photogrammetric framework, and then on using this knowledge within existing algorithms and designing new approaches that fully exploit these information sources.

### 2.1 Current Photogrammetric Capabilities

The photogrammetric capabilities of the Digital Mapping Lab are based on an object-oriented photogrammetry package, designed around standard photogrammetric models and written in C for compatibility with existing code.

The structure of the package is similar to that described in [McGlone, 1992]. Points and images are objects, implemented as structures, while a world object contains ellipsoid parameters and various control variables.

Images may come from sensors with various geometries; models currently implemented include frame, orthoimages, and multispectral line scanners. Each image object supports a set of methods, including the primary projection routines `image_to_world` and `world_to_image` and auxiliary methods which perform input and output, calculate partial derivatives, and other operations.

A central component of the photogrammetry package is the simultaneous multiple-image bundle adjustment program. The solution can involve images of differing geometries, such as frame and line scanner, or can use object space geometric constraints for additional information. All input parameters are weighted and full error propagation is performed on all output quantities. The generation of covariance information on derived parameters has proven to be nearly as important as the geometric calculations themselves, since it allows the specification of meaningful search bounds and more meaningful combination of different data sets.

A growing set of application routines computes image properties such as vanishing points, edge azimuths, and epipolar geometry. Applications are written around image methods, allowing them to be used for any sensor geometry that supports the operation in question.

### 2.2 Algorithmic Applications of Photogrammetric Techniques

Nearly all current areas of research in the Digital Mapping Laboratory use the photogrammetry package. This section will highlight the contributions of photogrammetric techniques to some of our current research areas, particularly results that could not have been obtained without this knowledge.

#### 2.2.1 Building Extraction

Our previous building extraction system, BABE [McKeown, 1990], was similar in some ways to most other building extraction systems in that it was designed to work only with vertical aerial images, and used only image space (pixel) measurements and reasoning. The incorporation of photogrammetric techniques allowed us to use BABE's algorithms in object space, permitting the generation of 3-D building models in world coordinates. A detailed discussion of this work is in Section 3.

#### 2.2.2 Stereo Processing

The existence of resection information for stereo pairs enables us to compute images in precise collinear epipolar alignment. Our experiments have shown that this gives better results than image warping approaches, especially for oblique stereo pairs [McKeown and others, 1993].

Since we have known image orientations, we can work in actual elevations instead of relative disparity. This allows us to easily evaluate the output against ground truth and to merge the results with other algorithms that also operate in object space (Section 4, [Roux and McKeown, 1994a; Cochran, 1994a; Cochran, 1994b]).

#### 2.2.3 Structure from 3-D Feature Matching

This work is predicated upon the ability to use the relative geometry between multiple overlapping sets of images to reduce the search space for feature matches (Section 4.3). Geometric information used includes epipolar line constraints, height range relative to a DEM, corner direction consistency, and line labeling (vertical-horizontal-neither) consistency.

Once image corner features in two images are matched, the image rays are intersected to generate 3-D corner points. To match a feature in subsequent images, the distance from the image ray through the feature to the hypothesized 3-D corner position is calculated. However, the precision of the 3-D corner position can vary greatly, depending upon the relative geometry of images determining it. Therefore, the distance calculation is done using the Mahalanobis distance, which essentially scales the physical distance according to the propagated covariance matrix of the point.

### 2.2.4 Site Modeling

A site model is a cartographic product; as such, it should be produced with methods that guarantee the best models, and generate quantifiable measures of the model quality. The use of photogrammetric methods allows the simultaneous use of all views of an object, as well as information such as right angle corners, etc., in determining object parameters. Stochastic information on input quantities can be propagated through to output results, allowing a quantitative evaluation of the site model produced, and also indicating what measures would be most effective in improving the model.

We currently have two methods for manual site model generation, differing in the implementation of the building models. Both are based on a simultaneous bundle adjustment, incorporating image orientations and ground points along with the building points.

The first method uses simple geometric building models, tied together to form more complex shapes if necessary. The parameters of the building models (length, width, etc.) are part of the simultaneous solution, and as such, have covariance information automatically calculated. A shortcoming of this approach is that the specification of the points, models, and connections involved can be complicated for complex buildings.

The other building modeling method uses point-wise constraints instead of explicit building models. Instead of constraining building points to lie on a rectangular prism, the points at the corners are constrained to form right angles and the building edges are constrained to be vertical or horizontal. This method allows easier solutions for buildings with complex shapes and simplified interactive specification of constraints.

Both methods generate covariance information on the building points. For the point-wise constraint method, building parameters and their covariances must be generated in a post-processing step if desired, since the solution yields only point coordinates.

### 2.2.5 Simulation

Nowhere are cartographic capabilities more important than in simulation. The end product **is** the representation of the environment; without consistent, realistic representations across all databases and participants, a simulation would collapse. To contribute to a virtual world, the results of image analysis algorithms must be presented in world coordinate frames, consistent with existing data for the simulation arena.

Our work in this area has involved efficient representations of terrain using Triangulated Irregular Networks (TINs) and the merging of extracted features into the simulation database [Polis *et al.*, 1994].

## 2.3 Future Efforts

Our preliminary efforts on the integration of photogrammetry and computer vision have proven fruitful; many of the following sections of this overview illustrate the effectiveness of photogrammetric modeling for augmenting performance of computer vision algorithms. In the future we plan to further study the fusion of data and results from multiple images, especially images from different sensor types. Central to this will be use of rigorous precision and reliability statistics to guide matching and evaluation. Effort will also continue in deriving vision cues for use in feature extraction algorithms.

We believe that only by merging the outputs of multiple algorithms can robust results be obtained; it is clear that only by doing this merging within a rigorous photogrammetric framework can practical solutions be obtained.

## 3 Automated Building Extraction in Object Space

Automated feature extraction from aerial images is a complex problem, and a variety of algorithms and techniques have been proposed to address this problem [Nicolin and Gabler, 1987; Huertas and Nevatia, 1988; Mohan and Nevatia, 1989; Irvin and McKeown, 1989; Liow and Pavlidis, 1990; McKeown, 1990; Fua and Hanson, 1991; Shufelt and McKeown, 1993; Lin *et al.*, 1994]. Many of the techniques currently employed in this domain make simplifying assumptions to produce building hypotheses. Two frequent assumptions are nadir acquisition geometry and orthographic projection. The former allows a building extraction algorithm to ignore potential building-building occlusions, and the latter allows right angles in the scene to be treated as right angles in the image (under the added assumption that building structure is comprised of right angles).

However, these assumptions place substantial restrictions on the kinds of imagery that can be handled. In oblique imagery, building-building occlusions are common, and walls of structures are visible. Further, the assumption of orthography is not valid even for nadir aerial mapping photography, as perspective effects are clearly visible where lines converge on vanishing points in the image. For a building extraction algorithm to be successfully used on a wide variety of imagery, it must be able to handle images from nadir and oblique viewpoints, and it must assume perspective projection to accurately model building structure in object space.

Our work in this domain has been focused on the integration of photogrammetric techniques into an existing building extraction algorithm [McKeown, 1990]. In previous work, the ability to generate 2-D image space building boundaries was demonstrated for both nadir and oblique imagery [McGlone and Shufelt, 1993]. In recent work, the building extraction algorithms have been extended to produce 3-D object space wire-frame buildings, measured in metric units. The resulting algorithm is capable of extracting 3-D buildings from a single view of an image, using the resection information provided by rigorous photogrammetric modeling [McGlone and Shufelt, 1994].

In the following sections, we discuss recent research on monocular building detection, including a brief discussion of the 2-D image space building extraction algorithm, and a detailed discussion of recently developed 3-D object space building extraction techniques. We present results for test scenes taken from the Fort Hood imagery distributed under the RADIUS program, as well as quantitative performance evaluation against manually compiled ground-truth building models, also made possible by object space representation. We also briefly demonstrate the ability to merge the results from multiple views in a common object space representation, as well as the ability to merge stereo-derived height estimates with monocular building boundaries.

### 3.1 Image Space Building Extraction

In earlier work, we developed techniques for incorporating vanishing point information into an existing building detection algorithm to produce 2-D image space building boundaries. Using the known image orientation, we hypothesize the orientation of image lines as horizontal, vertical, or "neither" lines in object space. This information is then used to constrain corner detection and building facet generation, and provides a means for handling oblique image geometries. In the remainder of this section, we briefly describe this approach, as it forms the basis for the 3-D object space building extraction algorithms. More detailed descriptions of the image space algorithm can be found in [McGlone and Shufelt, 1994].

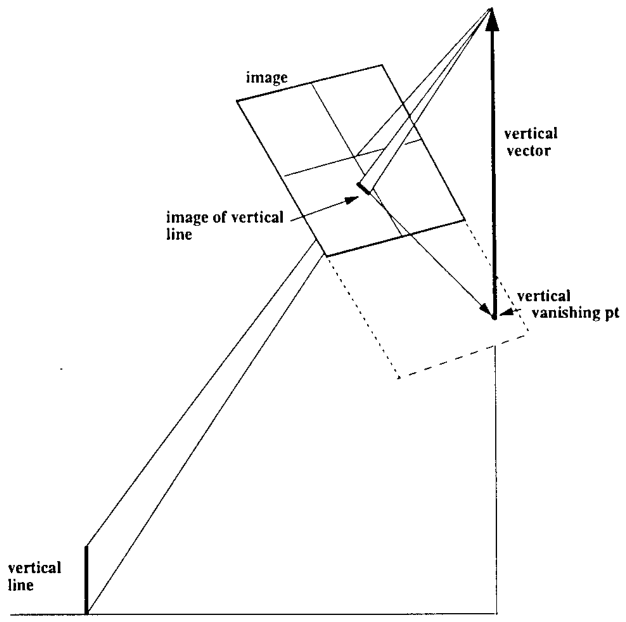


**Figure 1. Vertical vanishing point geometry**

### 3.1.1 Vanishing Points for Line Labeling

As is well known from projective geometry [Barnard, 1983], parallel lines in a scene meet at a common point in the image of the scene. This point is known as a *vanishing point*, since it is the image of a point at infinity on the parallel lines. This apparent convergence of parallel lines gives important cues to the orientation of the image and to the structure of objects within the scene. Figure 1 illustrates the geometry of vertical vanishing points.

Given $M$, the ground-to-image orientation matrix, and $v$, a vertical vector in object space, $Mv$ gives the vertical vector transformed into the image coordinate system. When placed at the perspective center, this vector pierces the image plane $z = 0$ at a point, which is the vertical vanishing point.

Once the vertical vanishing point has been computed, vertical edges can be identified by fitting each edge to a line constrained to pass through the vertical vanishing point. If the RMS error of the residuals exceeds 2.0 pixels, the line is not labeled as a vertical. An unconstrained line is also fit to each remaining edge; if the slope of this edge differs from the slope of a line passing through the center of the edge and the vanishing point by more than 0.2 radians, the edge is not labeled as a vertical. All remaining lines are labeled verticals.

To find the horizontal vanishing points, each edge in the image is assumed to be horizontal in object space, and the azimuth of each edge is computed. These azimuths are then histogrammed. Under the assumption that man-made structures are defined by perpendicular sets of parallel lines, the azimuth histogram is examined for mutually supportive sets of perpendicular lines. Instead of selecting the histogram bin with the maximum score, the score of each bin is added to the scores of the bins representing perpendicular directions. The maximum of this sum represents the strongest mutually perpendicular sets of parallel lines in the scene, and the lines that contributed to these bins are labeled horizontals. In areas where buildings and roads lie on a common grid, one maxima is sufficient; in areas where buildings are oriented in a variety of directions, secondary maxima can be examined.

Figure 2 shows an oblique image of a barracks area in Fort Hood, Texas. Figure 3 shows the edges

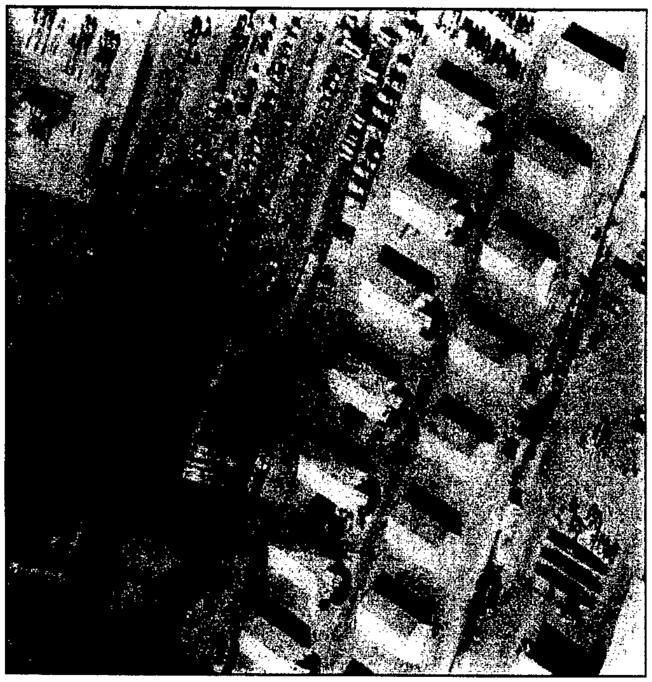


**Figure 2.** RADT9WOB **test scene**

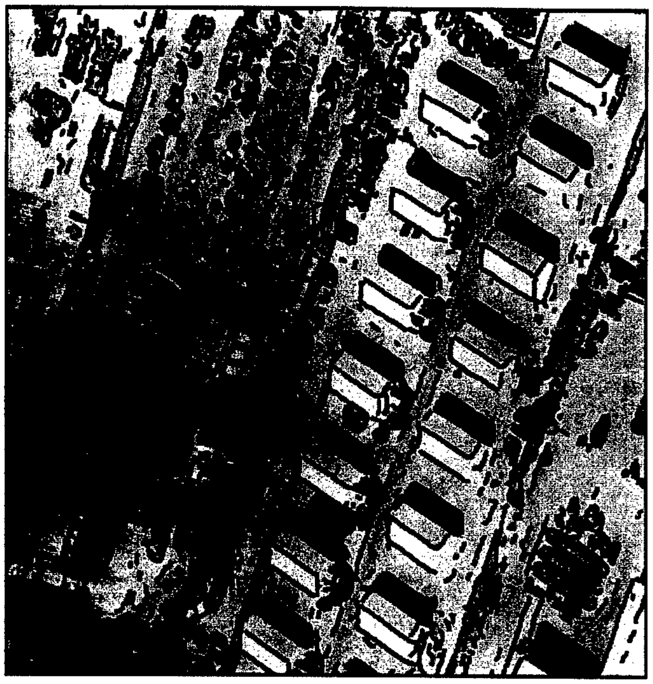


**Figure 3.** RADT9WOB **edges**

extracted from this image by an implementation of the Nevatia-Babu line finder [Nevatia and Babu, 1980]. Candidate verticals and horizontals are shown in Figures 4 and 5. Some edges are labeled as both horizontal and vertical because of the viewing angle of the image. In these cases, other information or multiple views must be employed to decide on the correct labeling.

### 3.1.2 Corner and Box Detection with Horizontal and Vertical Line Attributes

Under the assumption that man-made features in aerial photography can be modeled by parallelopipeds joined at edges, horizontal and vertical edge labelings can be used to assemble building hypotheses. In a standard approach, a range search is used to find lines whose endpoints are in close proximity and hence form corners. These corners are then linked by their edges to form boxes, or building facets [McKeown, 1990].

The line labelings can be used to prune the set of corners under consideration, and they can also be employed as a geometric constraint for boxes generated by the corner linking phase. This is done by using a simple building model, as shown in Figure 6. Each distinct line segment in the diagram has been assigned a label, indicating its orientation in object space: horizontal, vertical, or neither.

From this diagram it is apparent that only certain combinations of labeled lines are allowable in forming corners and boxes. For instance, two intersecting verticals never form a valid corner in object space. As another example, two horizontals are only allowed to form a corner if their intersection in object space forms a right angle. An example of a box constraint is that vertical walls can only be formed by alternating vertical and horizontal lines.

These heuristics allow the set of corners and boxes to be constrained greatly, which cuts down on the later processing needed to verify hypothesized boxes. Figure 7 shows all possible boxes generated by the corner linking algorithm; there are 4,128 boxes. Figure 8 shows the subset of boxes that have legal

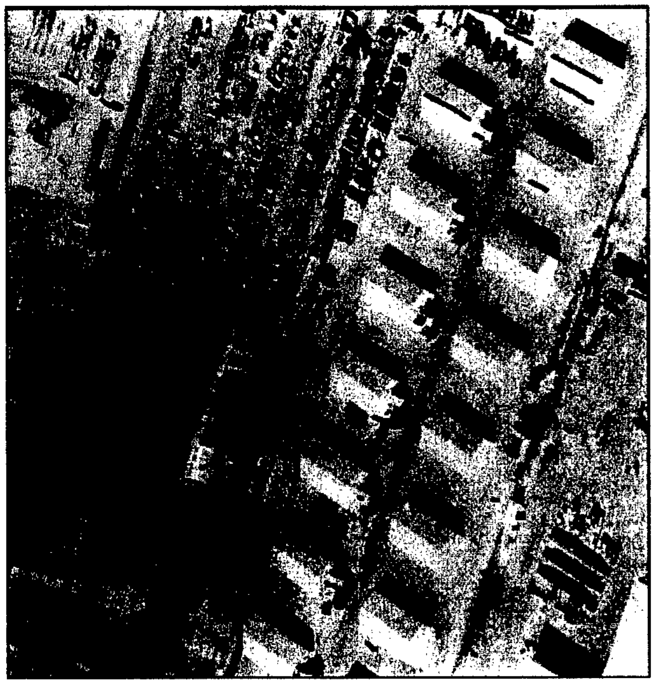


Figure 4. Vertical edges

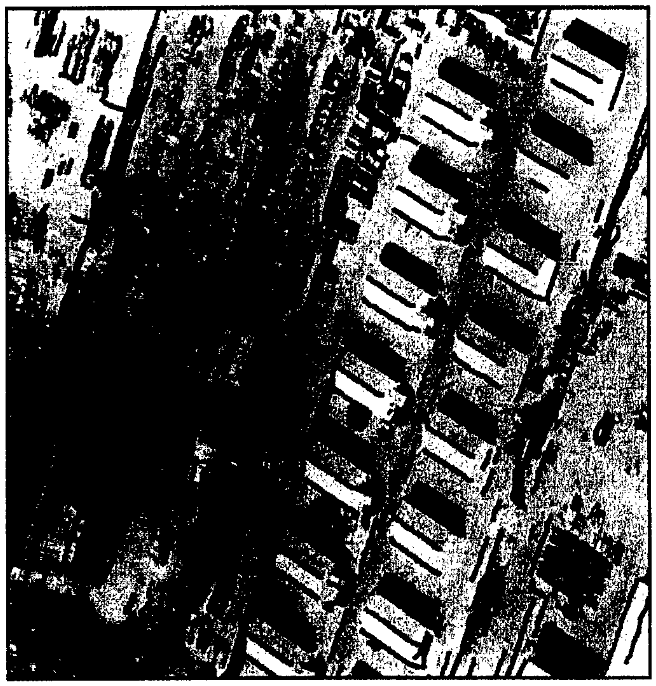
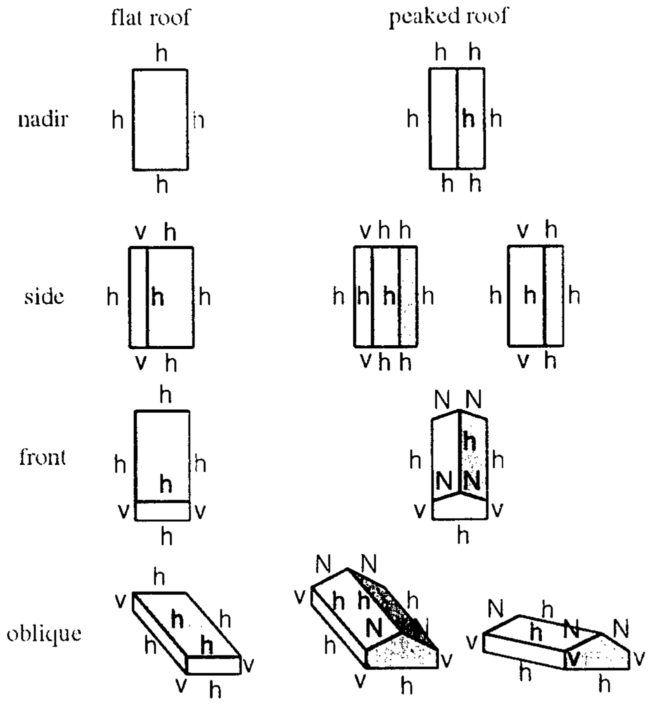


Figure 5. Horizontal edges



h - horizontal line
v - vertical line
N - unclassified line (neither horizontal nor vertical)

Figure 6. A simple building model

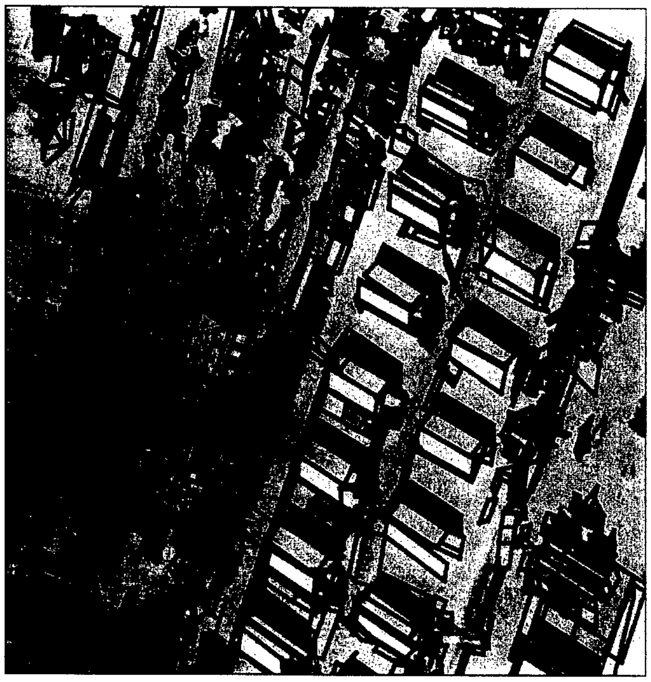


Figure 7. All hypothesized boxes

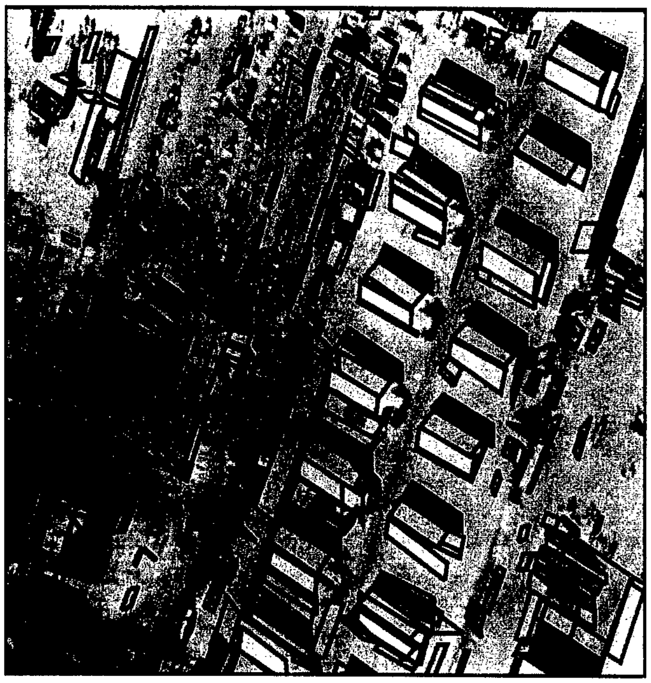


Figure 8. Geometrically consistent boxes

line labelings according to the building model in Figure 6; only 776 boxes are left after this geometric consistency check.

Once geometric constraints have been used, photometric constraints are employed to verify building hypotheses; in the current system, each box is examined for the presence of a shadow, under the nadir acquisition assumption, and does not make use of the photogrammetric information. We intend to address this shortcoming in future work.

### 3.1.3 Peaked Roof Extrapolation

It is rarely the case that all possible building facets are generated from the image data, because of missing or noisy edges, corners not found during range search, and search heuristic failures during box construction. However, given a set of verified building hypotheses, the line labelings on the boxes can be used to hypothesize the presence of missing box facets. We have implemented one such technique, known as peaked roof extrapolation.

Figure 9 illustrates the situation at hand. The hypothesized facet represents a 2-D box hypothesis that we wish to use as a guide for hypothesizing the other half of the rooftop. We begin by computing the line perpendicular to the horizontal line $R$ in object space, and projecting this perpendicular into image space (line $C$). Next, we intersect that line with the line drawn through the roof peak point $p$ and the vertical vanishing point $vp$, to obtain a point $x$. In object space, the distance between $x$ and $e$ is equal to the distance between $x$ and $n$; we assume that these distances are equal in image space as well, and complete the new building facet by using the roof peak point $p$, points $n$ and $f$, and the application of symmetry to generate $g$.

Figure 10 shows the verified set of box hypotheses. Figure 11 shows the results of applying the peak projection technique to the boxes with an alternating horizontal-neither labeling. This approach allows

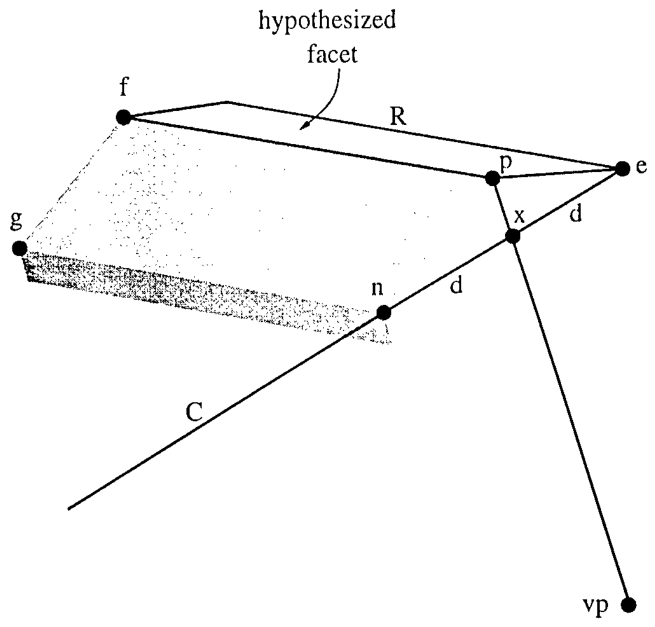


**Figure 9. Peaked roof projection**

failures in shadow verification and edge detection to be addressed by the use of projective geometry and photogrammetric information.

## 3.2 Object Space Building Hypothesis Generation

In this section, we consider the problem of determining the height of 2-D building hypotheses from a monocular view. Previous research in this area has typically involved some form of shadow mensuration, by associating dark regions in the image with building hypotheses and measuring their lengths in image space [Irvin and McKeown, 1989]. Such measurements have typically used approximations to the sun elevation angle in order to estimate structure height from shadow length, again producing a height estimate in terms of image space units.

Image space-based shadow mensuration techniques encounter difficulties in the oblique domain. In nadir photography, shadows are adjacent to the structures casting them, making the association of shadow regions with building hypotheses a relatively easy task. Under wide angles of obliquity, however, it is difficult to correctly associate shadow regions with roof regions without a boundary estimate of the wall to link the two, which is essentially what we seek when attempting to derive roof height.

These techniques also encounter difficulties that are independent of the acquisition geometry. Approximations of the sun elevation angle can introduce substantial error in height estimates, depending on sun location at the time of image acquisition. Difficulties also arise in measuring the length of a shadow in image space; the dark shadow regions often have noisy boundaries, which could be due to noise in the image, occluding objects on the ground, or changes in ground elevation.

An alternative approach is possible under photogrammetric control, using our simple building model. Given roof hypotheses, we can search for vertical lines in image space at roof corner points, and measure the heights of these verticals in object space to obtain height estimates for the roof. In the next two sections, we discuss issues in reliable location of vertical lines at corner points, and methods for using these lines to measure heights for flat and peaked roof buildings.

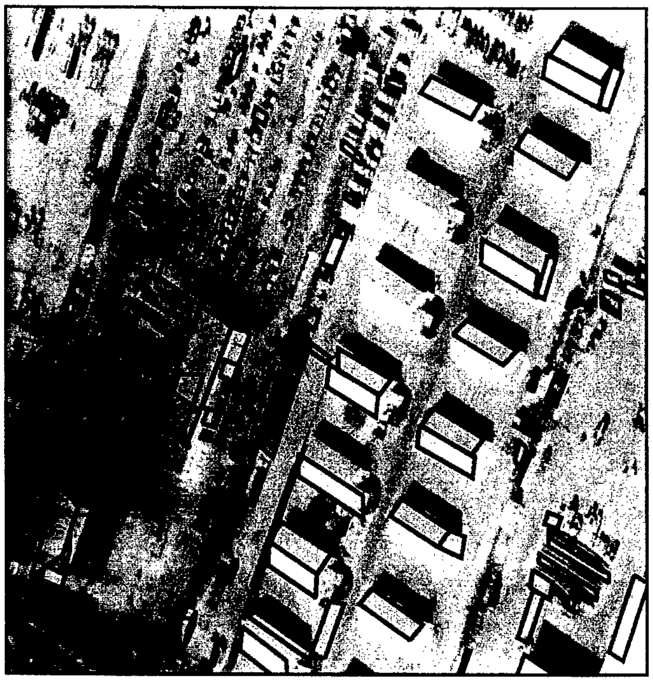


**Figure 10. Before peak projection**

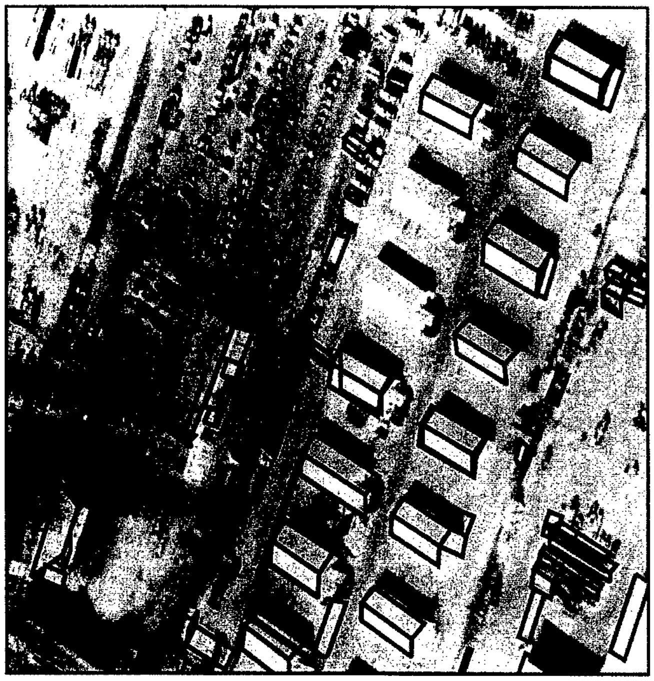


**Figure 11. After peak projection**

### 3.2.1 Vertical Line Location

The goal of vertical line location is to find a vertical edge in image space that emanates from a specific point. In our case, we wish to find vertical lines at roof corner points, under the assumption that such lines must constitute the edges where building walls meet.

A simple way to find such verticals would be to return to the original edge data and use the corner points as a basis for range search to find edges with vertical labels. This approach, however, is susceptible to the quality of the edge data, which can be poor for vertical edges. While template-based edge detectors perform reasonably well on long straight lines in aerial photography, they tend to round edges at corners, and often do not locate the vertical edges, which are typically much shorter. This means that potential vertical edges are often mislabeled as horizontals or "neither" edges because of the rounding at corners, which can alter the computed orientation of the edge, or, the potential vertical segments are not separated from other edges because of their shortness, and instead, are misinterpreted as noise at the end of an edge segment.

To avoid these difficulties, we instead focus processing attention on the corner points, which are likely starting points for any vertical lines, and we use oriented edge-finding techniques to maximize the likelihood of finding short edges. We now outline the vertical line finding strategy we have developed, which performs well in finding short vertical edges.

We use an imperfect sequence finding technique [Aviad, 1988] to locate a line of pixels, beginning at a corner point, and oriented in the direction of the vertical vanishing point, which have gradients higher than a certain threshold in the direction perpendicular to the line. Starting with the corner pixel, each pixel is tested to see if it has sufficient gradient support in the direction perpendicular to the line to be labeled an edge point.

This labeling process produces a binary sequence of points, which are either labeled as edge points, or non-edge points. The imperfect sequence finder is used to locate the terminating point of this sequence, which will be the other endpoint of the vertical line. The sequence finding technique is used for two reasons; first, to tolerate noise along the potential vertical line, and second, to handle the potentially noisy gradient values in the immediate vicinity of corners, where many edges may meet.

The edge/non-edge determination for each pixel is carried out by locating gradient extrema inside a window around the pixel, fitting a line to these extrema, and computing the residual error of this line with respect to the extrema points. A confidence score, weighting in the residual error of the fitted line and its slope, with respect to the vertical vanishing point line, is computed. If this confidence score passes a threshold, the pixel is labeled an edge point; otherwise, it is labeled a non-edge pixel. This scheme allows correctly oriented lines with noisy gradient to be tolerated, since the slope of these lines will be close to that of the vanishing point line; it also allows for slight orientation errors to be tolerated if gradient support is high, when a line fits the gradient extrema well.

In practice, given a corner point, the vertical line finding process is invoked from each pixel in a window around the corner point to produce a set of possible verticals for each corner. This is done to alleviate the problem of corner localization; because of edge noise or line fitting errors, corners are not always well localized at the corner points. To select the best vertical from the set, a confidence score is computed for each vertical line in the same fashion as the confidence computation for pixels, except that the evaluation window covers the entire line. The vertical with the largest product of length and confidence is then selected as the most likely vertical for the corner point.

Figure 12 shows the final set of verticals produced by the vertical line finding process for the RADT9WOB scene. Comparing this result with the original vertical line detection result in Figure 4, it is clear that guided edge extraction from seed corner points provides an improved method for locating vertical lines. The area surrounded by the small square in this figure is shown in closer detail in Figure 13. This example shows the set of verticals grown from points in a 1-pixel radius around the corner point of a peaked roof facet; roof facet line labelings are denoted by H for horizontal lines and N for "neither" lines. The darkened vertical is the one ultimately selected from the set as the best vertical, based on the length and confidence scoring.

### 3.2.2 Height Estimation

If we assume that a given edge represents a vertical line in the scene and that the elevation at the bottom of the line is known, we can calculate its height using similar triangles, as shown in Figure 14. We first calculate the coordinates of the bottom point, $P_1$, and the top point as if it were at the same elevation as the bottom, $P_2$. $D_1$ is then the distance between points $P_1$ and $P_2$, $D_4$ is the distance from the image to $P_2$. $D_3$, the ground distance between the point $P_2$ and the point directly below the image, is then calculated using $D_4$ and $H$, the height of the image above the elevation of $P_1$:

$$D_3 = \sqrt{D_4^2 - H^2}$$

The height of the object $h$ is then, from similar triangles,

$$h = H\frac{D_1}{D_3}$$

After applying the vertical line finder to each corner point of every 2-D building hypothesis, we measure each vertical line in object space to obtain a height value, producing height estimates at every corner of every hypothesis. Corners with verticals contained within the 2-D hypothesis boundary are not used for height estimation, since these vertical lines would not be visible in the image, and hence, are artifacts of the vertical line finding process. Many of the verticals in Figure 12 fall into this category.

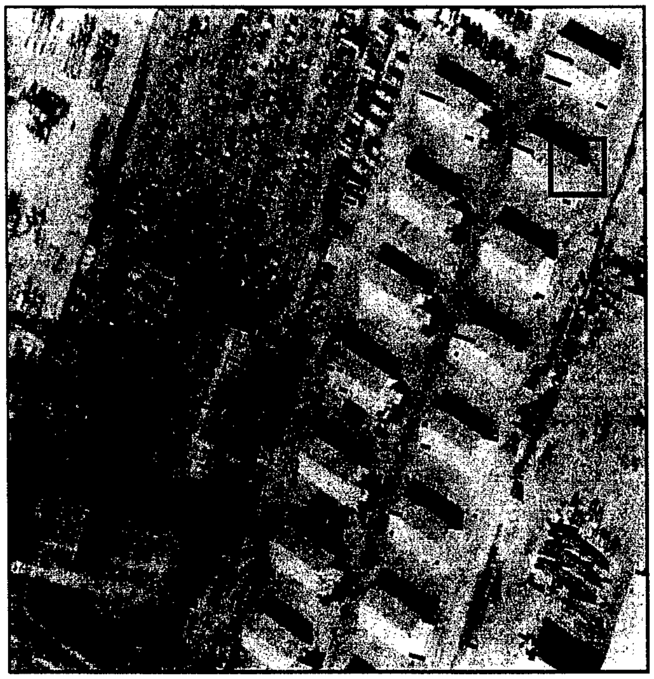


Figure 12. All verticals

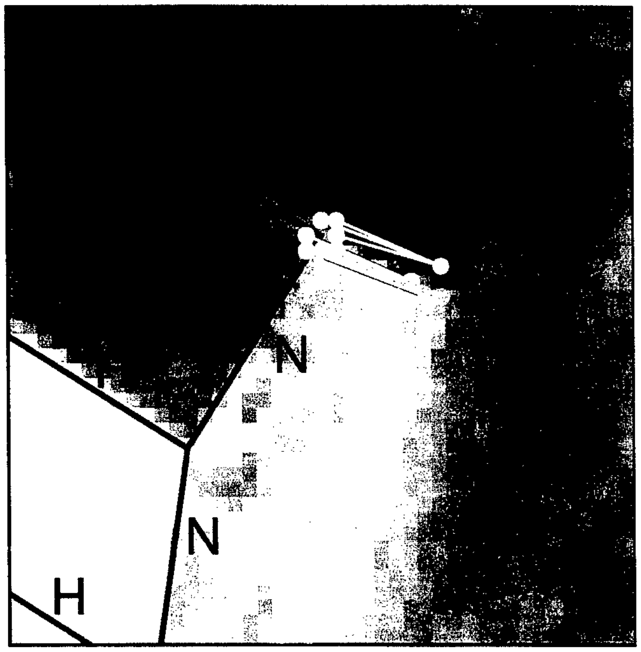


Figure 13. Verticals at one corner

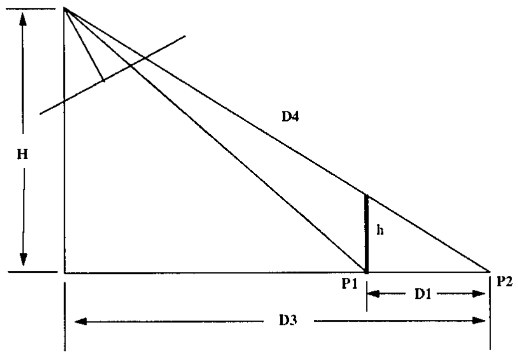


Figure 14. Geometry of height estimation

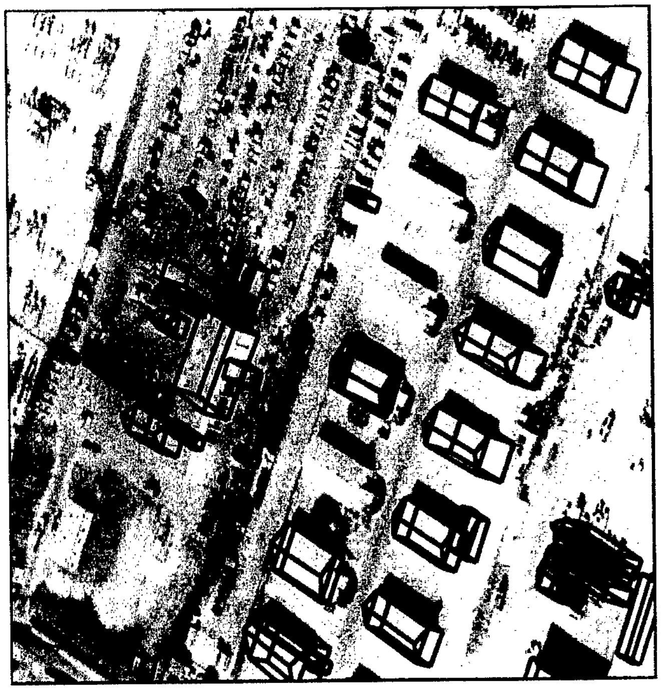


**Figure 15.** RADT9WOB **final results**

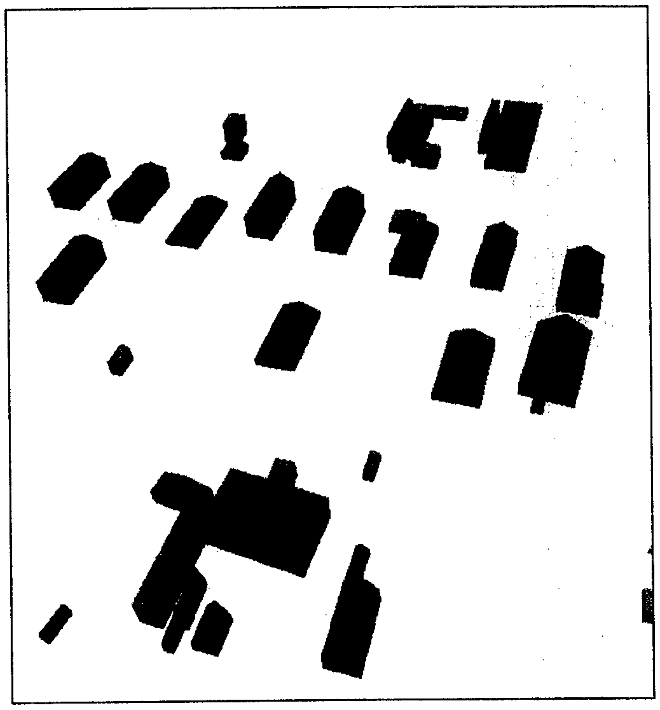


**Figure 16. Perspective view**

Of the remaining verticals for each hypothesis, one is chosen with the largest product of length and confidence score, and the height associated with this vertical is used as the height for the entire structure. Since vertical edges are typically extracted shorter than they really are, the longest strong vertical is expected to be the most reliable. For flat roof buildings, this nearly completes the 3-D hypothesis process; all that remains is to project the 2-D hypothesis into object space using the height estimate, and to construct a 3-D wire-frame model by dropping points from the 2-D boundary points. Ground elevation, of course, must be derived by some other means; for our experiments, we indexed into a DEM of the Fort Hood site to obtain the local ground elevation for each 3-D structure.

A similar process is used for peaked roof buildings to obtain the height of the flat portion of the peaked structure. It remains, however, to compute the height of the peak above the imaginary flat roof line. This can easily be performed by using information extracted during the peaked roof extrapolation phase. Returning to Figure 9, we note that $p$ and $x$ form a vertical line, which we measure in object space to obtain the height of the peaked portion of the structure. The absolute height of the peak is then computed by adding the flat height estimate to this peak height estimate.

With object space measurements of each building structure, we perform a pruning step to weed away implausible buildings. Currently, any structure less than two meters in length, width, or height is pruned, but these can of course be modified to suit the typical buildings expected in the scene. In previous implementations, pruning mechanisms, such as these, were based on ad-hoc image space thresholds, which could be related to actual object space properties only through implicit assumptions about image scale and acquisition geometry.

Figure 15 shows the object space models generated by this technique for RADT9WOB, projected back into image space. Figure 16 shows a perspective rendering of these models, and illustrates the 3-D capabilities of this extraction system. The structures shown here have heights ranging from 2 meters, the pruning threshold, to 13.8 meters. These heights are qualitatively comparable to those measured manually. We discuss quantitative performance in the next section.

We note that while shadow analysis was not used for height estimation in this work, it still constitutes a valuable source of information. In future work, we hope to integrate shadow analysis and vertical line finding to provide more reliable estimation of structure height; by using verticals to guide the search for shadows, the difficulties mentioned earlier can be alleviated.

## 3.3 Performance Evaluation

In the following sections, we discuss our strategy for quantitative evaluation of the performance of these building detection techniques. In Section 3.3.1, we describe our approach for generating ground truth models of the test scenes, for image space and object space comparisons; we also define evaluation metrics for capturing system performance. In Section 3.3.2, we present results for a test scene visible in two nadir and two oblique images of the Fort Hood site, and analyze the results.

### 3.3.1 Evaluation Methodology

Evaluation of the test results was done against a manually-generated model of the buildings in each test scene, using monocular measurements of building corner points in all images covering the scene. A simultaneous photogrammetric bundle adjustment was done for each test scene which included the measured points on each building, the original control points, and all four images of the scene. As part of the solution, building points were constrained to fit the specified type of building model (flat or peaked roof). The use of a simultaneous adjustment incorporating the building geometric constraints insures that the most consistent and accurate building estimates are obtained.

The imagery used was provided as part of the RADIUS program; a complication in the adjustment and in the later processing was the fact that it was geometrically processed to simulate an unspecified sensor. We approximated the unknown sensor using a frame camera model, which provided a reasonable fit across the image but had residual parallax in some of the test areas. To prevent this unusual situation from biasing the processing and evaluation we treated the bundle adjustment of each scene as dealing with separate images, and used the orientation information from the adjustment for each scene in processing that scene. In effect, this approximated the geometry of each processed image with piecewise frame images.

For evaluation purposes, we use scene-wide metrics that analyze the degree of overlap between the automated results and the manually-generated models. These metrics allow us to treat extraction errors of all types in a uniform way, and provide an unbiased measure of system performance. These metrics also have the advantage of being applicable in both 2-D and 3-D, allowing quantitative comparisons of 2-D building detection and delineation performance with the height estimation performance in 3-D.

In image space, we regard an automated extraction result as a classification of each pixel in the image as either building or non-building. An overlap comparison is then simply a pixel-by-pixel comparison of the 2-D projections of the results of the automated system against the 2-D projection of the manually-generated building models. Measurements in image space allow us to assess the delineation capabilities of the system.

In object space, we regard an extraction result as a classification of regions of space as either building or non-building. An overlap comparison in this domain can be implemented as a voxel-by-voxel comparison of the 3-D models generated by the automated system and the 3-D manually-generated models. Measurements in object space allow us to assess the height estimation capabilities of the system.

Each overlap comparison produces a count of true positives (both manual and automated results detect building), false positives (the automated result shows a building, while the manual result does not), and true negatives (the manual result shows a building, while the automated result does not). We define four metrics from these pixel/voxel counts:

- *detection percentage* = $\frac{100TP}{TP+TN}$. This metric measures the percentage of building pixels/voxels in the manual results that were actually detected by the automated system.
- *branch factor* = $\frac{FP}{TP}$. This metric, proposed in [Shufelt and McKeown, 1993], measures the degree to which the automated system "over-hypothesizes" building structure.
- *miss factor* = $\frac{TN}{TP}$. This metric, the counterpart of branch factor, measures the degree to which the automated system fails to hypothesize existing building structure.
- *quality percentage* = $\frac{100TP}{TP+FP+TN}$. This metric summarizes overall system performance. Any false positives or true negatives are reflected in this score, and will lower the quality percentage.

Before proceeding to the quantitative evaluations of the automated results, we note that all 3-D overlap comparisons were done by first discretizing object space into voxels at 0.5m resolution, and then comparing the manual and automated results at each voxel in object space. This is approximately the ground sample distance of the Fort Hood imagery and was deemed sufficient to provide reliable quantitative evaluation.

### 3.3.2 Experimental Results

Our experimentation has been limited to five test areas visible in each of four images of Fort Hood. Two of the images have near-nadir geometry, while two are oblique. The scenes contain a variety of building structures, ranging from simple flat roof and peaked roof buildings, to L-shaped structures and buildings composed of multiple rectangular volumes.

For brevity, we will only consider one test area in detail, the RADT5 scene. Figures 17 and 19 show RADT5 and RADT5S, two near-nadir views of barracks in Fort Hood. Figures 18 and 20 show two views of the same barracks at varying degrees of obliquity. Superimposed on all four images are the final results of the building extraction process, the 3-D models generated in object space and projected back into image space.

We first consider the image space overlap statistics in Table 1, presented in the first four columns of the table. The building detection percentages for RADT5 and RADT5OB are quite high, indicating that much of the building structure was detected. For the other two scenes, the percentages are much lower, because of failures in different processing phases. In RADT5S, the initial set of hypothesized boxes cover most buildings in the scene, but the scene is closer to the vertical vanishing point than RADT5. This fact combined with the lack of contrast leads to poor vertical finding, even with the application of the oriented line finding technique described in Section 3.2.1. Hence, many of the boxes have very low computed heights, and are pruned away.

In RADT5WOB, the few boxes that are correctly delineated in image space obtain good height estimates from the vertical line location process. In most cases, however, the boxes generated by BABE are either poorly delineated, due to missing edges along the road behind the barracks; or they are false hypotheses, formed by alignments with road horizontals and roof horizontals. The 2-D verification process currently performed by BABE rejects many of the poorly delineated boxes, even though they do cover many peaked roof facets.

The image space statistics reflect this performance; the miss factors for RADT5 and RADT5OB are low, indicating good performance in locating building structure. The branch factors for both of these scenes

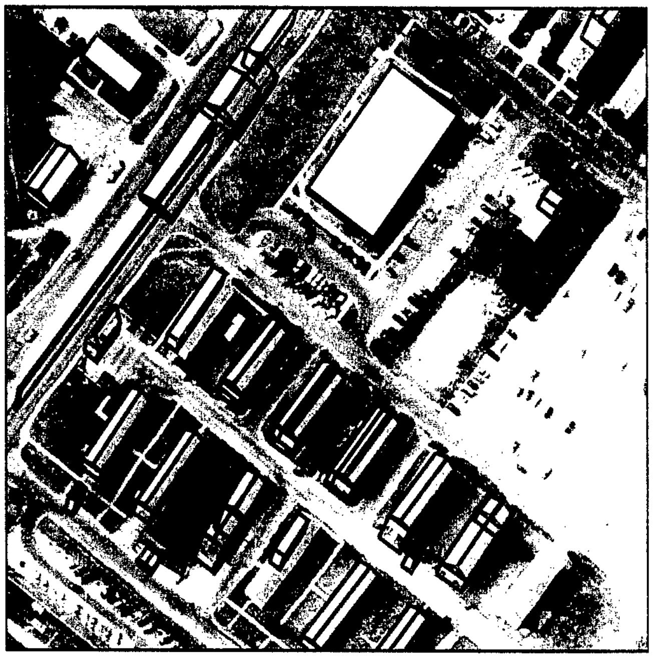


Figure 17. RADT5 results

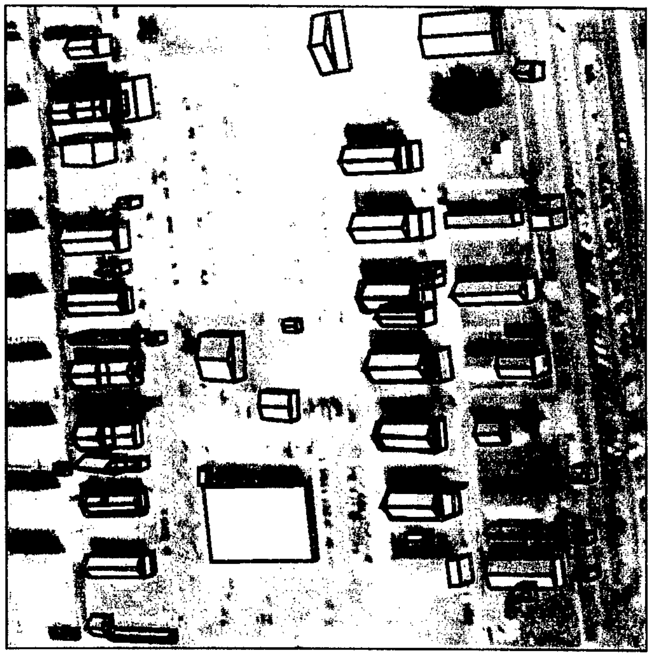


Figure 18. RADT5OB results

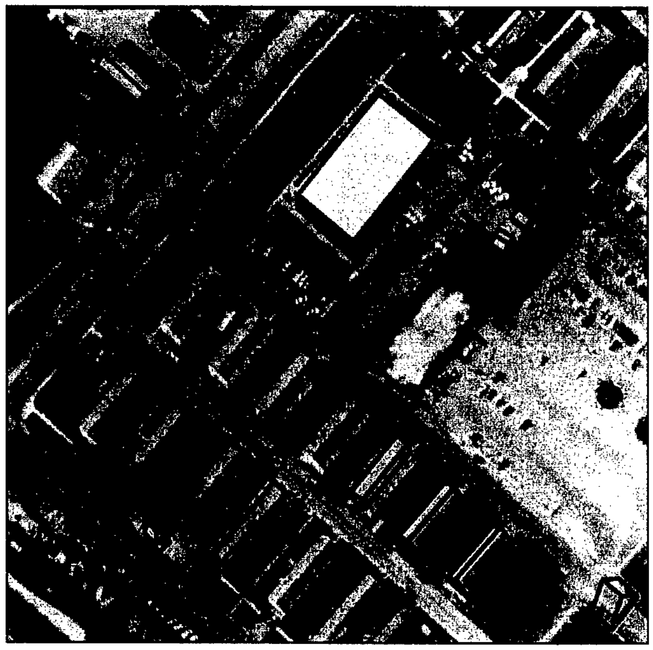


Figure 19. RADT5S results

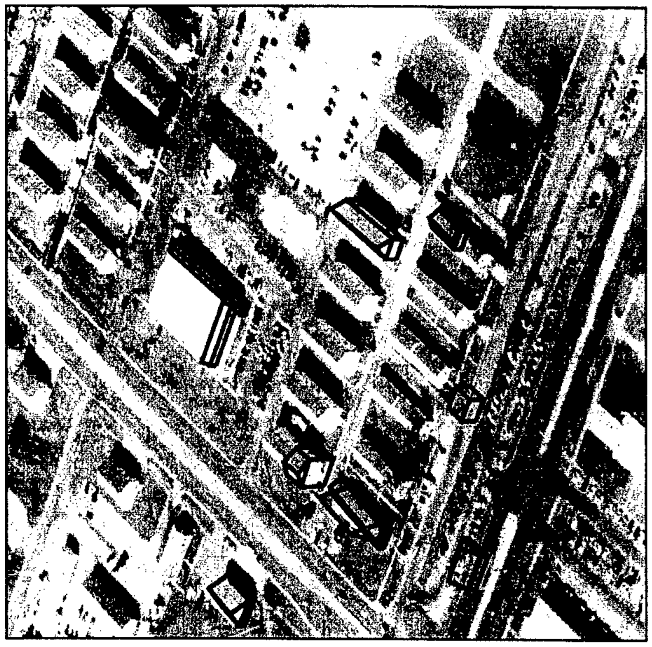


Figure 20. RADT5WOB results

**Table 1. Image space statistics,** RADT5 **scene**

| | bld | brf | miss | qual |
|---|---|---|---|---|
| RADT5 | 80.2 | 0.56 | 0.25 | 55.3 |
| RADT5OB | 82.0 | 0.84 | 0.22 | 48.5 |
| RADT5S | 36.9 | 0.80 | 1.71 | 28.5 |
| RADT5WOB | 24.5 | 0.63 | 3.08 | 21.2 |

**Table 2. Object space statistics,** RADT5 **scene**

| | bld | brf | miss | qual |
|---|---|---|---|---|
| RADT5 | 41.5 | 2.22 | 1.41 | 21.6 |
| RADT5OB | 54.7 | 1.85 | 0.83 | 27.2 |
| RADT5S | 21.7 | 2.01 | 3.60 | 15.1 |
| RADT5WOB | 10.1 | 1.94 | 8.93 | 8.4 |

are higher, indicating that more false positive structures were hypothesized for these scenes as well. The miss factors for RADT5S and RADT5WOB show that much building structure is missed in these scenes; in the latter case, almost three times as much structure is missed than is detected.

The object space overlap statistics in Table 2 present a similar performance picture, although the relative scores in the four metrics are noticeably worse. This is to be expected, since heights are derived from typically short vertical lines, and errors in vertical line extent on the order of a pixel can translate into height errors of a meter or more in object space. Nonetheless, the same performance trends can still be observed in object space; in RADT5 and RADT5OB, the miss factors are relatively low. The lowest miss factor is produced by RADT5OB, which illustrates the improvement in height estimation when strong verticals are present in the image at object corners.

In both image space and object space evaluations, the quality scores are low, despite the good qualitative performance on RADT5 and RADT5OB. This is to be expected as well; the quality metric treats true negatives and false positives with the same weighting as true positives, and thus, is very sensitive to error. From a pixel classification standpoint, such a metric may be regarded as overly harsh; in fact, if we count the number of correctly classified pixels in the image and divide by the total number of pixels in the image, we find that the four scenes have classification rates of 85 percent to 91 percent. We believe, however, that this type of classification metric is inadequate, because of its insensitivity to error. Many urban and suburban scenes are composed of small fractions of building pixels; a system that hypothesized no structure whatsoever in these scenes would receive a high classification score, although its qualitative performance in building detection would be poor. The quality metric does not suffer from this flaw.

We have observed similar performance trends in the other test areas in Fort Hood. When vertical lines are prominent and boxes are reliably hypothesized in image space, building extraction performance is relatively good. In other cases, a combination of complex building shapes and poor contrast at building edges causes substantial difficulties for the box hypothesis mechanism, and final performance is very poor. Most of these difficulties, however, rest in the image space hypothesis generation and verification phases, which remain topics of current work.

We conclude our analysis with a detailed discussion of two example buildings from the Fort Hood test scenes, both of which exhibited poor quantitative and qualitative performance. The examples we present here show common causes of detection failures, and many of the failures we have observed across our test dataset are caused by the problems we describe below.

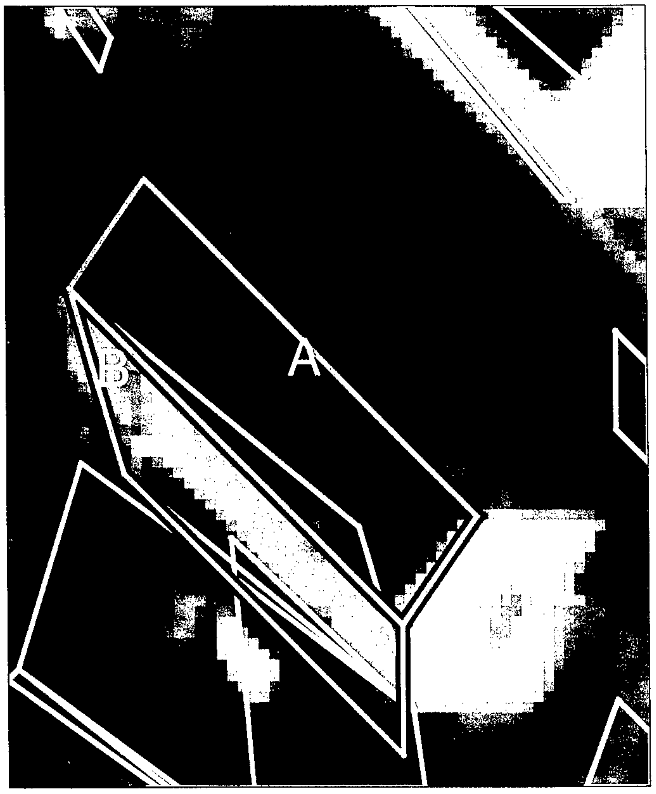


**Figure 21. Initial box hypotheses**

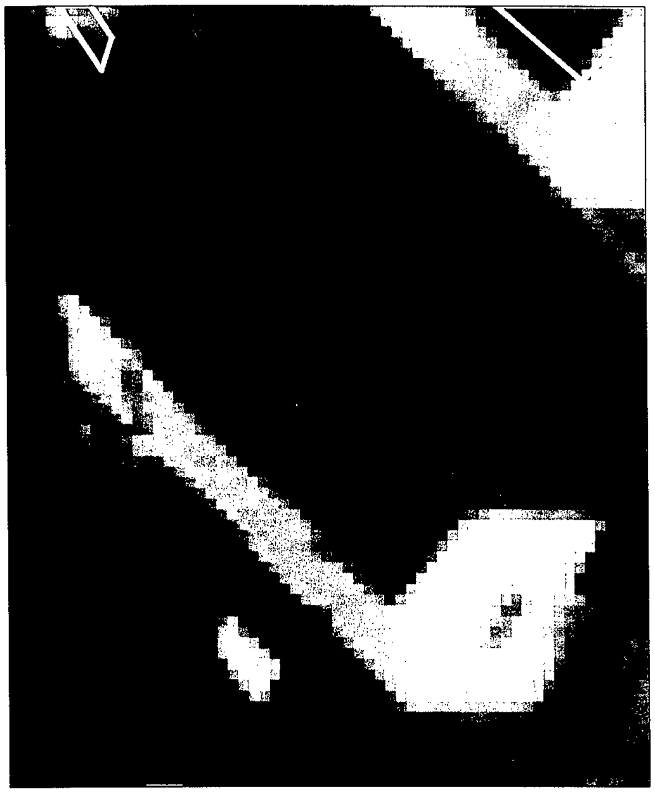


**Figure 22. Verification failure**

The current system employs an image space shadow verification algorithm, which assumes only flat roof buildings and a nadir acquisition geometry. Although the algorithms we have described often perform well when these assumptions are violated, in many situations the result is the rejection of many valid roof facet hypotheses. We turn to one such example from RADT5WOB, in which many peaked roof buildings were undetected.

Figure 21 shows one of the peaked roof buildings in RADT5WOB, along with the boxes generated by BABE for this piece of image. These boxes passed the geometric consistency phase; *i.e.*, the labelings assigned to them by vertical and horizontal analysis were consistent with the allowable facets for building models. We will focus our analysis on boxes A and B in the picture. Figure 22 shows the boxes remaining after image space shadow verification; neither A nor B were verified.

Box A failed verification because of a violation of the nadir-acquisition assumption. The image space verifier treated A as a flat roof, and examined the expected shadow casting edge for a transition from light to dark, indicating a possible shadow region. It found a lighter region on the expected shadow casting edge, and rejected A, when in fact this lighter region was a wall of the structure and was adjacent to the true shadow region.

Box B failed verification in a more indirect fashion, but one which still highlights the need for true object space modeling and verification. The image space verifier computes a shadow threshold by sampling intensities near supposed shadow-casting edges for all boxes, histogramming these values, and adaptively selecting a threshold that cuts off at the darkest peak in the histogram. B does have the expected light to dark transition on its expected shadow-casting edge, but the intensity inside this region (which is really Box A) is not in the darkest peak of the histogram, which corresponded to shadows associated with structures that were correctly detected in RADT5WOB.

These examples clearly demonstrate the necessity of true 3-D object verification which takes into account scene geometry and illumination. In these cases, the low-level facet generation phases provided reasonable seeds for further processing, which were not exploited because of faulty assumptions in verification. In many other situations, however, the low-level facet generation algorithms are the root of detection failures. We now turn to an example in RADT11 that illustrates several low-level failures that must ultimately be addressed in future work.

Figure 23 shows an L-shaped building in RADT11, with the edges extracted for this portion of the scene. Figure-ground contrast is good for this building; however, the garage entrances on the vertical wall with upper edge C cause severe fragmentation in the edges extracted by the edge finder. Problems of this nature occur in images acquired at nadir, but they are worsened in oblique views since walls often have entrances, windows, and other textural features that can cause increased fragmentation effects.

Figure 24 shows the line-corner graph generated by BABE for this scene, with the only two boxes (D and E) actually generated for the underlying L-shaped structure. Other boxes generated outside the building were omitted for clarity. Given the lines seen here, BABE would be expected to generate two boxes, one for each wing of the building, since it is based on creating boxes from corners, and does not attempt to model composite shapes. Instead, even though some of the lines (notably the shadow-side lines) were extracted with little fragmentation, the box generation heuristics fail to generate boxes that completely cover both wings. These heuristics are designed to start at a corner in the graph and find the closest right angle in the graph to create a box. Both D and E are closed prematurely because of this heuristic. D is closed midway down the wing because of a dark feature on the ground that forms an accidental corner; E is closed immediately because of the extensive fragmentation along C, which produces many false corners. These problems, common in many test images, demand more robust heuristics and techniques for box generation, independently of the verification and object modeling problems outlined earlier.

These examples illustrate the need for true 3-D modeling of object structure. Ideally, the generation and verification algorithms would work with three-dimensional models in object space, rather than 2-D boxes in image space. This strategy would allow all feasible models to reach verification, where precise geometric information permits rigorous testing of illumination constraints across adjacent planar surfaces, prediction and verification of cast shadows [Irvin and McKeown, 1989], and the application of stereoscopic information for consistency constraints across multiple views. Understanding these issues and the development of rigorous techniques to address these problems, as well as improving the performance of low-level hypothesis generation algorithms, are the subjects of current research.

## 3.4 Summary of Building Extraction

We have described experiments in incorporating photogrammetric calculations in an existing building detection system, analyzed the results on a small set of nadir and oblique aerial images, and raised several issues of modeling, hypothesis generation, and hypothesis verification that must ultimately be addressed in a complete implementation of a photogrammetrically rigorous feature extractor. We have presented qualitative and quantitative results for nadir and oblique imagery that show that the combination of precise camera modeling and geometric information with existing feature extraction algorithms provides a powerful approach for increasing the performance of building detectors on complex aerial imagery.

The integration of photogrammetric information into a building extraction system augments the capabilities of the system along several dimensions. It provides a mechanism for metric measurements of image features; provides useful cues for building extraction, such as the horizontal and vertical line labelings used throughout this work, and establishes a common object space representation for reasoning about structures and combining information from multiple views.

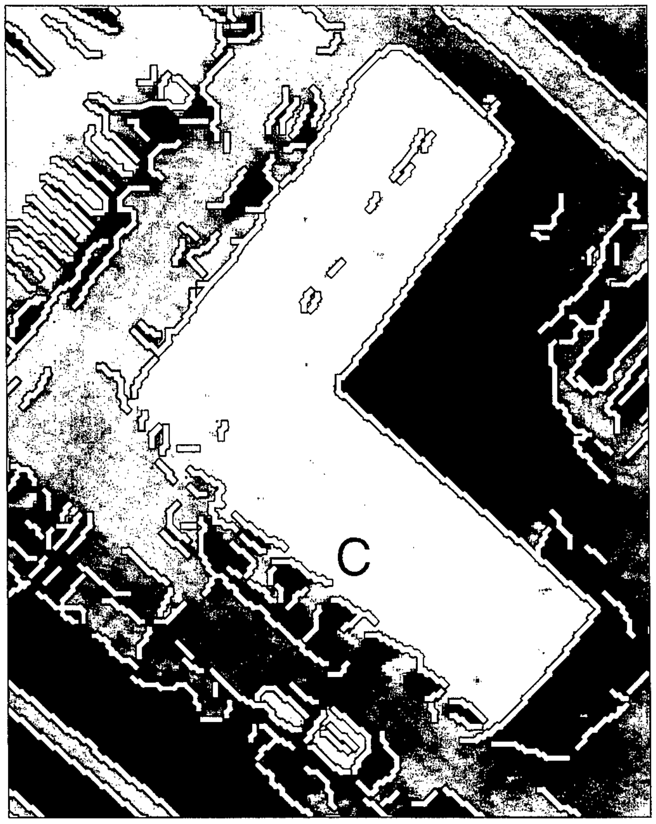


Figure 23. Fragmented edges

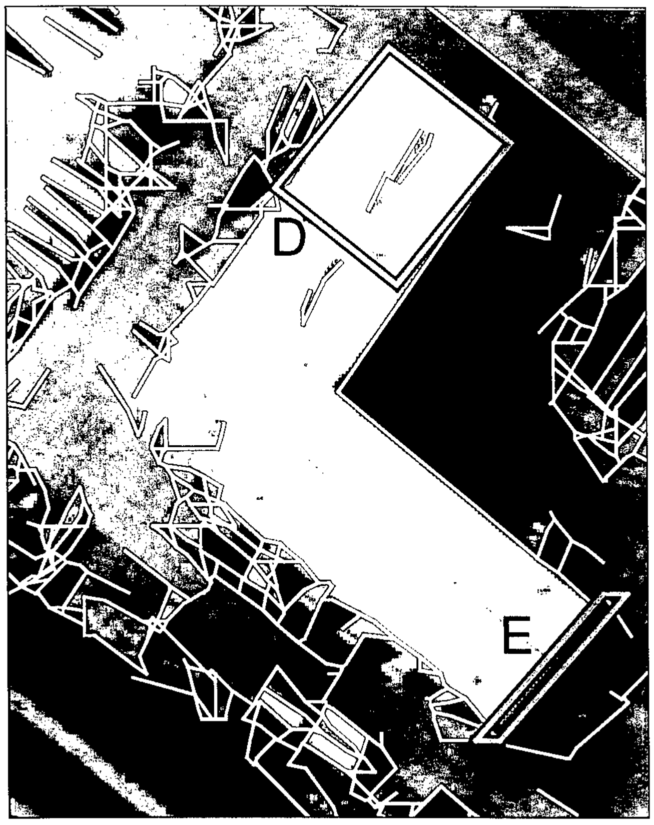


Figure 24. Line grouping errors

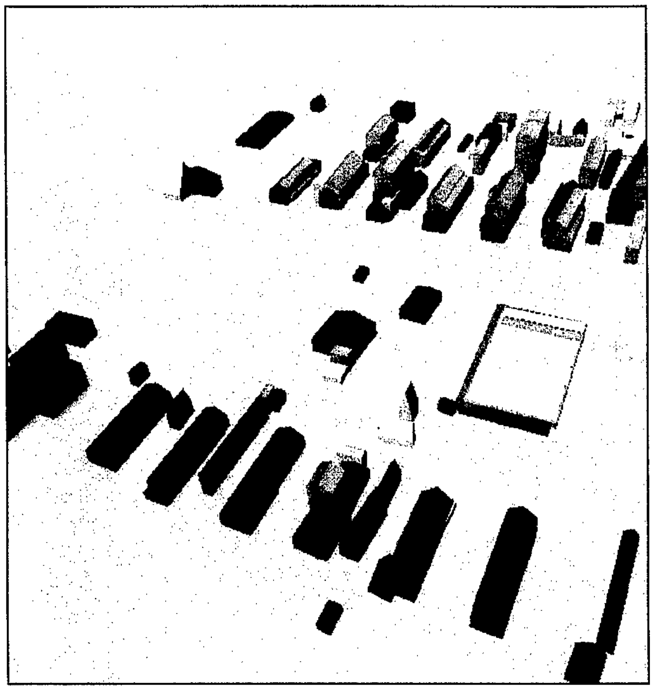


Figure 25. Multiple results in object space

Figure 26. RADT5OB results with stereo elevation

The use of a common object space representation opens up new possibilities for improving building detection performance. Given multiple views of a scene, the results of building extraction algorithms can be placed in a common object space, where further analysis can take place, and information fusion techniques can be employed. Figure 25 shows the results in Figures 17–20, in object space. Each set of results from the four test areas is shown in a different shade of grey. The building extraction algorithm derives complementary estimates of building structure from each image; given these estimates, information fusion techniques could be employed to produce building hypotheses consistent across several views [Shufelt and McKeown, 1993].

Figure 26 shows the result of merging elevation estimates generated by an adaptive vergence stereo process [Cochran, 1994a; Cochran, 1994b] with the 3-D object space structures produced by the monocular building extractor. The buildings have been placed on the underlying DEM. This illustrates the utility of rigorous photogrammetry for object extraction; given a common world representation, it is possible to merge structural information from a variety of sources to produce improved estimates of building structure. This will prove essential for producing digital building models of a quality suitable for inclusion in simulation databases [McKeown and Lukes, 1992; Polis *et al.*, 1994].

## 4 Research in Stereo and Multiple Image Analysis

We are currently working in three areas related to stereo matching and the incremental analysis of multiple views of a scene. The first is directed toward understanding the strengths and weaknesses of our existing stereo matching systems and to provide remedies to improve overall performance. Toward this end, we have integrated object-space knowledge into the stereo process to adaptively adjust the stereo search matching region. The second part explores a new area-based stereo approach that is based in object-space using multiple (more than two) images. The third examines object space feature matching for building delineation across multiple-images.

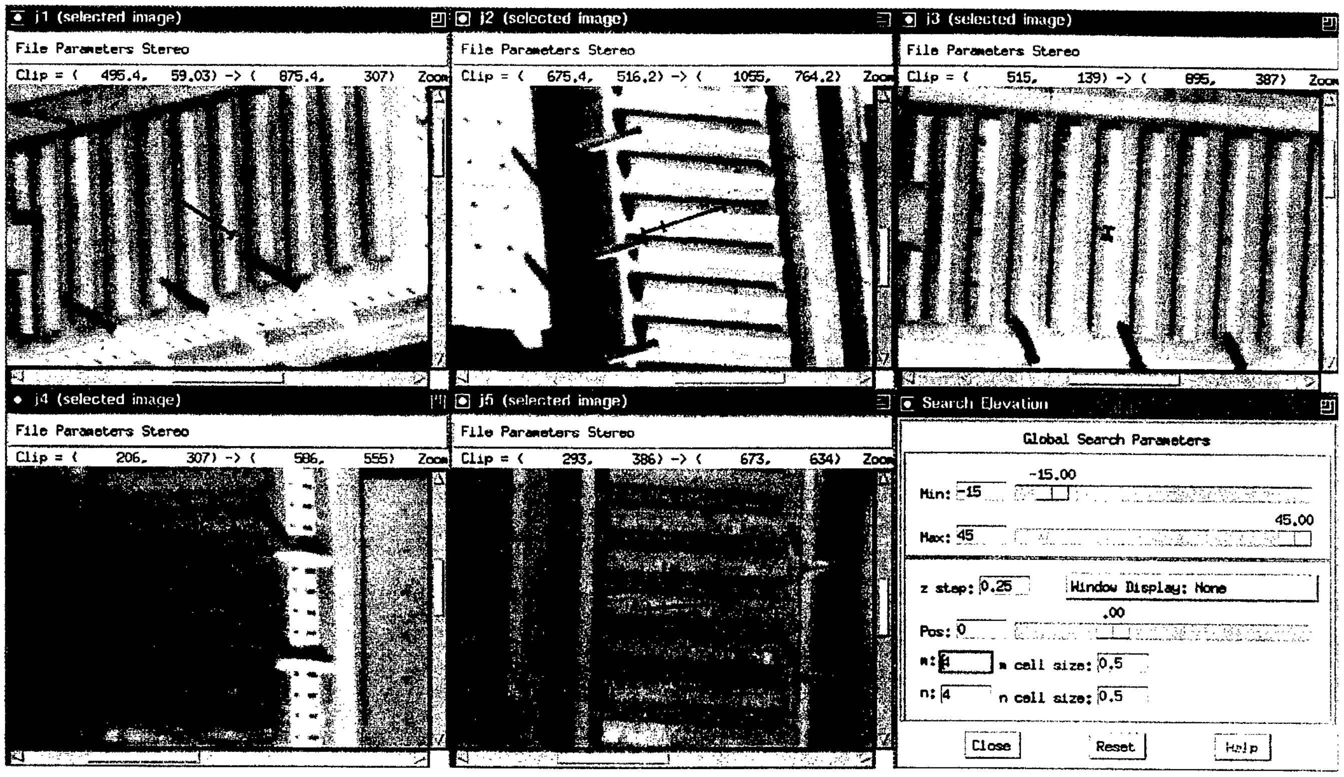
### 4.1 Adaptive Vergence

The use of oblique imagery for the extraction of man-made features, such as buildings for cartographic applications, introduces several problems to the process of stereo matching that are not present when traditional near-nadir imagery is used. When oblique imagery is brought into a collinear epipolar alignment, the ground-plane is usually not perpendicular to the camera axis, a simplifying assumption made with most near-nadir imagery. In addition, walls are now commonly visible rather than being exceptional features. These problems require changes to the processing of the imagery and in the operation of the stereo matching.

The obliquity of the ground-plane causes the most problems for stereo matchers. To accommodate the larger range of disparity, the stereo process must either increase its search window, or use an adaptive vergence mechanism to control the search window on a pixel-by-pixel basis. Increasing the search window increases both the search time and the chance for aliasing to occur. We have shown [Cochran, 1994a] that adding an adaptive vergence mechanism to an existing stereo system provides a substantial improvement in its ability to process oblique imagery. A more detailed description of this current work can be found in [Cochran, 1994b].

### 4.2 Object Space Based Stereo

We are extending the idea of working in object-space as opposed to image space in a new approach to area-based stereo matching which allows 2–N images. The key idea in this research is to conduct the stereo



Clip = ( 495.4, 59.03) -> ( 875.4, 307)

Clip = ( 675.4, 516.2) -> ( 1055, 764.2)

Clip = ( 515, 139) -> ( 895, 387)

Clip = ( 206, 307) -> ( 586, 555)

Clip = ( 293, 386) -> ( 673, 634)

Min: -15 (-15.00)

Max: 45 (45.00)

z step: 0.25 Window Display: None

Pos: 0 (.00)

m: 4 m cell size: 0.5

n: 4 n cell size: 0.5

**Figure 27. Epipolar search-range between a set of images of the area-of-interest and a hypothetical nadir view**

search by selecting an initial vergence and evaluating potentially matching pixels in the set of images by projecting a match window into each of the images and locating the best match (or matches) within the search range. The search range may be varied from the initial vergence position by moving the match window in object space, or by adjusting its slope and tilt, to match expected or hypothesized local surface conditions.

Figure 27 shows five images that cover the same area-of-interest (these are the RADIUS modelboard #1 images J1–J5). In our initial implementation of this approach, we select a point in any one image, which, along with a DEM of the region or an estimate of the ground from the resection control points, is used to specify the "ground point" in object-space. That is, the object-space coordinates of the intersection of the ray from the camera center through the selected pixel and the ground. The match window is placed tangent to the Earth's surface at this point and is projected into each of the images. This match window is allowed to move above and below the initial (zero) vergence "ground point" in fixed steps. In the figure, the search range is between $-5$ and $+45$ meters with 0.25 meter steps. Figure 28 shows a world view of this type of search arrangement. The match window follows the epipolar line between each image and a line through the "ground point" to zenith.

In Figure 29(a) we plot the correlation of a single area viewed from the nadir along the indicated search range in image J1 (upper left image in Figure 27). This gives us a close comparison with traditional area-based stereo matching. The differences are that we do not reproject the images into a collinear epipolar alignment, but travel along the epipolar projection, and that the match window is aligned with the object-space local-vertical coordinate system rather than with the epipolar projection. Two sources of error are present in the results. First, the aliased area along the search range (the peak of the adjacent vent) appears as a slightly better match for the point, and second, elevation of the correct match (the local

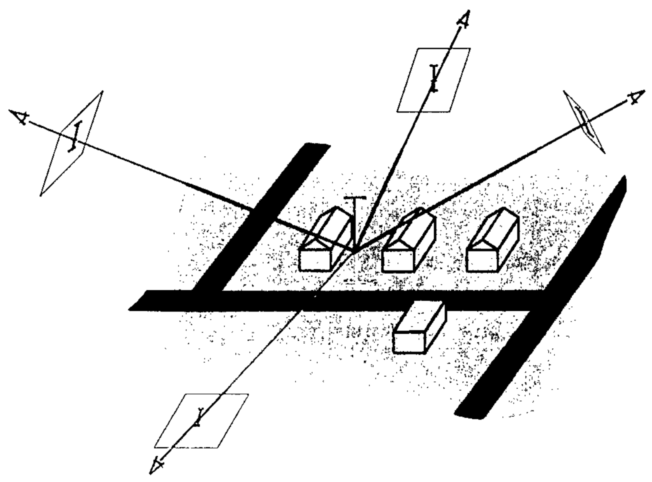


**Figure 28. World view of the search-range**

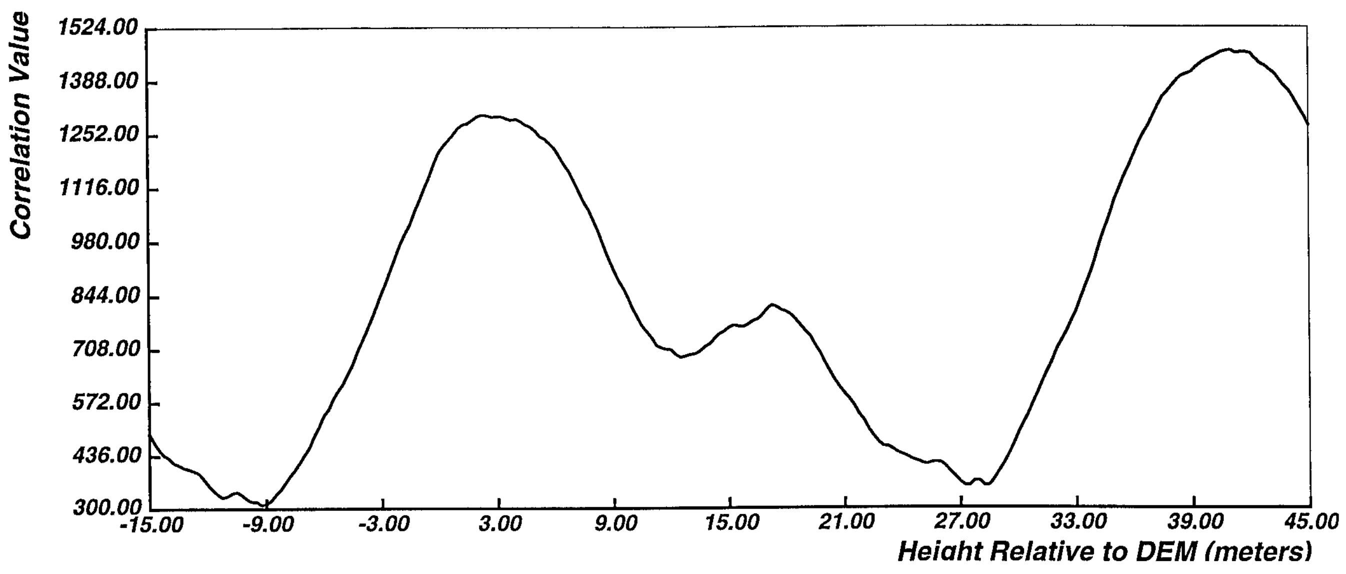
minima furthest to the left) is off from the 32.55 meter actual height of the selected point. This latter problem is mostly caused by small image-specific errors in the resection (see Section 2.1). The best human selected match using this single image gives a height of 28.99 meters and is within a pixel of the correct local minima found by the stereo process. In Figure 29(b) we plot a generalized multi-image correlation between all of the selected images by stepping along the search range in steps representing 0.25 meter elevations in object-space. For each point we plot the average of all pairs of two-image correlations. In this plot, we now see that the correct match has become the global minima and the projection into object-space is better localized with a height of 32.75 meters, an error of 0.20 meters or about $^1/_5$ pixel in image J1[1].

Some of the interesting attributes of this approach are that it automatically incorporates image warping [Quam, 1984; Norvelle, 1992] and adaptive vergence [Cochran, 1994a; Cochran, 1994b]. This is because the match window is projected into each of the target images from object space where, if there is sufficient *a priori* information such as a DEM or prior stereo estimate of the surface, the window will both track the ground surface, and may be fit to the local slope and tilt of that surface. Figure 30 shows one view of the projected match window from the above example.

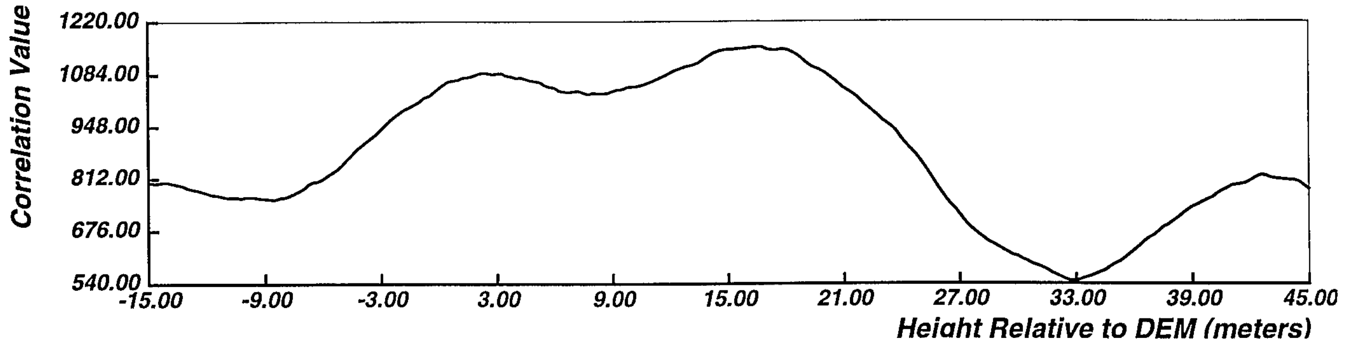
Pixel versus subpixel matching is now based on the projected size of the match window into each of the target images. Instead of setting a pixel size, we must now consider what to do when a cell of the match window is subpixel, when it spans multiple pixels and how to smoothly transition between these two extremes. We currently use a cubic interpolation when the cell size is smaller than a pixel, a weighted median average when the cell spans multiple pixels and a linear transition between these methods as the cell size grows between 1 and 2 pixels. In addition, as a side effect of using a photogrammetric approach to projecting the match window into the set of images, we can apply the stereo matching to images of different resolution. This latter effect may be of advantage as we begin using larger sets of images, covering a common area, which were not originally obtained as stereo pairs.

Since we are not comparing a single pixel in one image against a set of possible matches in another image, but rather a series of multi-pixel matches at different elevations, we have the possibility of having multiple matches. This occurs along vertical surfaces, such as building walls. We can use this feature along with hints provided from other analysis such as multi-view feature matching [Roux and McKeown, 1994a]

[1]A position shift in object-space of one meter in elevation causes an image shift in image J1 of 0.99 pixels (−0.41 rows, −0.90 cols).



(a) Area-based correlation along search range of image J1 with an incorrect global minima due to aliasing and a 3.56 meter error due primarily to the projection into object-space being based on a single stereo pair



(b) Area-based correlation of all five images over the same search range

Figure 29. Response from a correlation operator over the search range from Figure 27

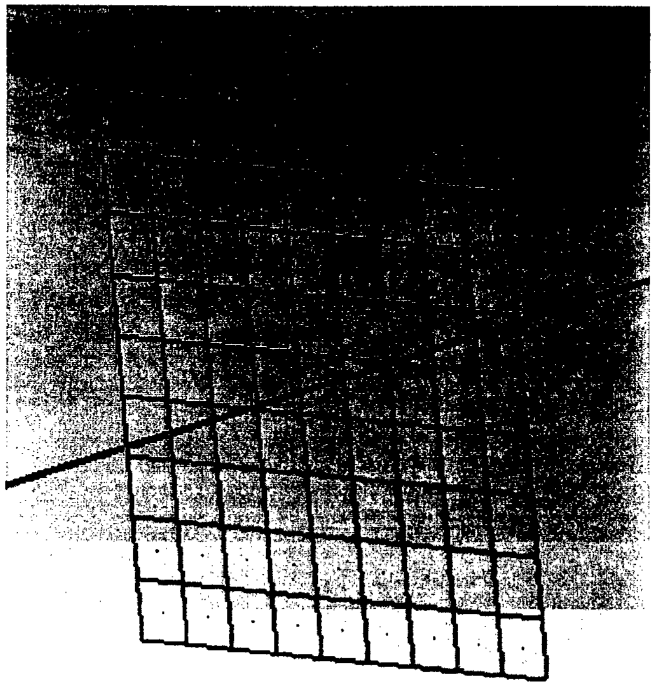


**Figure 30. Close-up view of the match window projected into image J2 along the epipolar line shown in Figure 27**

(see also, Section 4.3 below) or from monocular analysis [Shufelt and McKeown, 1993] to orient the match window vertically and along the expected wall surface and to prefer multiple matches while looking for the points where the matches break off.

Finally, as we search for the best cluster(s) of views, we can also look for outliers from this cluster. This is especially interesting when a tight cluster exists among some views and a common set of outlier views are present for neighboring points in object-space. Therefore, this method of stereo matching has the possibility of labeling some spatial points as being occluded from some of the selected image viewpoints. It also will give us some constraints on the possible elevations for a point visible from these images at the corresponding image pixel.

Future work is directed toward:

1. improving the matching by recognizing outliers in the clustering of the values in the match windows from multiple views of the scene because of occlusion, photogrammetric or radiometric distortions,
2. the detection and adjustment for local offsets in the photogrammetric resection which represent errors that were not modeled, and
3. integrating hints from other processes into the stereo matching process.

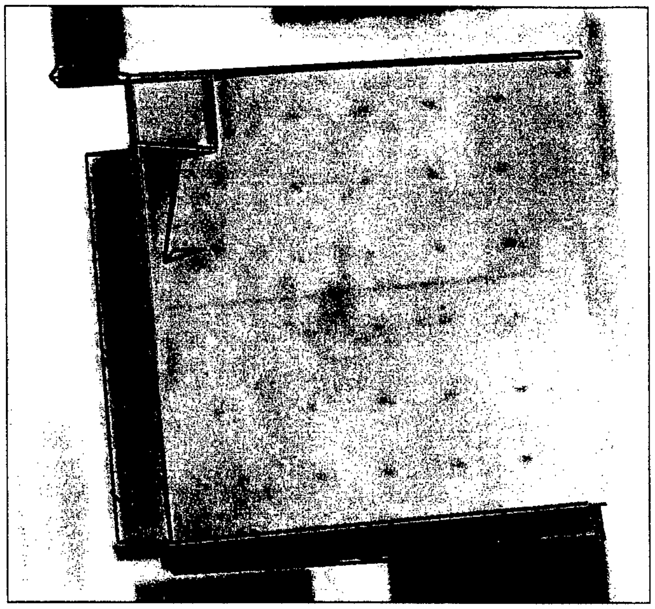


**Figure 31. Strong and weak segments from 3 views**

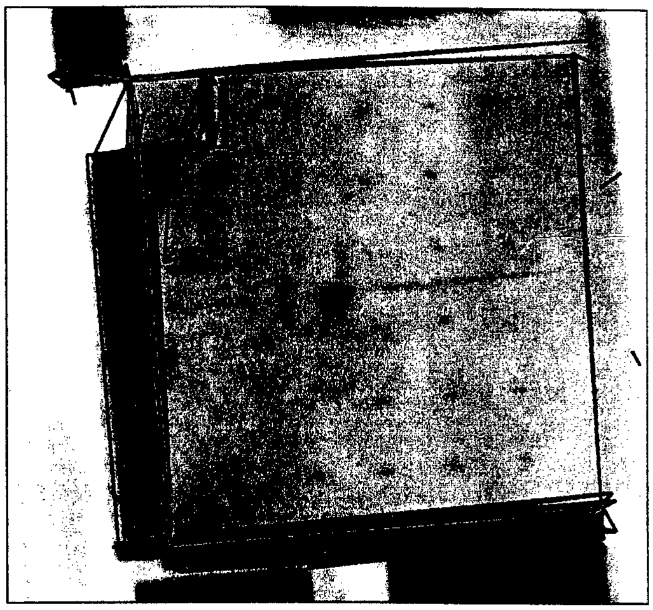


**Figure 32. Strong and weak segments from 4 views**

## 4.3 Feature Matching for Building Extraction from Multiple Views

In Section 3 we described our current progress in monocular analysis using photogrammetric constraints to generate building hypotheses. Coupled with traditional stereo analysis, this work provides a good starting point for detailed modeling of complex buildings and other man-made structures. However, in keeping with our philosophy of using multiple methods to improve overall accuracy, we are developing a new method for building extraction using multiple views and based on object space feature matching. Each new image viewpoint adds new object surfaces not previously seen and provides additional observations of existing surfaces. These additional observations add information since there are usually significant changes in illumination and viewing geometry. The accumulation of multiple views also reduces potential mismatching due to accidental alignments and increases the 3-D positioning accuracy of derived object models by simultaneous solution of collinearity equations. Within the general framework of analysis of multiple views, many plausible approaches can be investigated: pairwise structure combination, and simultaneous solution of the 3-D position of the features. The approach developed here is based on the successive incorporation of new image data into an existing partial solution. A detailed description of this work can be found in [Roux and McKeown, 1994b]. In the remainder of this section we briefly illustrate the results produced using this new method of object space matching of corners and line segments and their composition into 3-D surfaces.

Figures 31–33 show 3-D line segments (strong in red and weak in green) projected onto a small portion of a reference modelboard image, generated using three, four, and five views of the scene. As additional views are added, many redundant estimates are created for several of the occluding roof edges, and the rightmost edge is not hypothesized until the addition of the fourth view. Figure 34 shows the two 3-D surfaces that are created using the object space lines in Figure 33.

Figures 35 and 36 show the building roof surfaces hypothesized using three and five views, respectively. While this work is still somewhat preliminary and relies on an accurate photogrammetric model for all of the images, it clearly demonstrates that incremental modeling is possible and that the new information

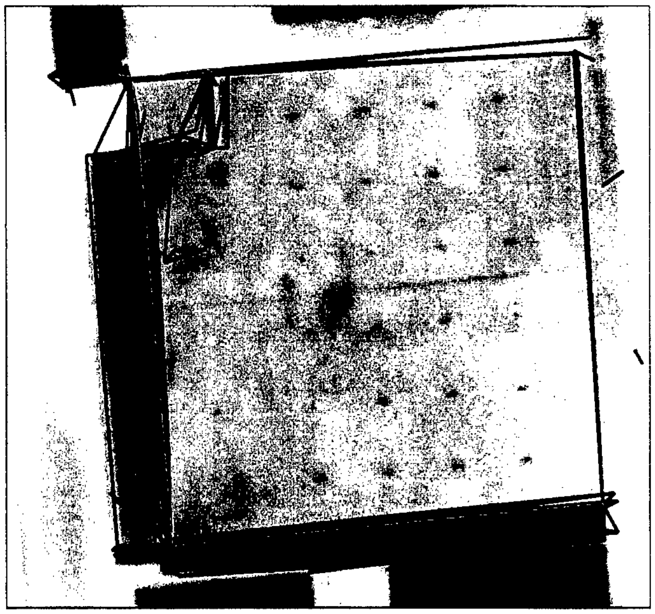


**Figure 33. Strong and weak segments from 5 views**

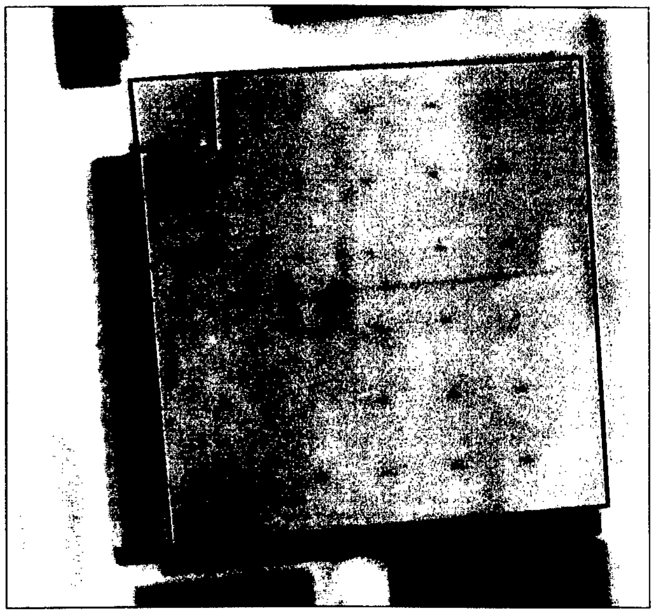


**Figure 34. Surfaces from 5 views**

provided by additional views can be integrated into a single consistent building representation.

## 5 Multispectral Analysis

With the availability of moderate resolution multispectral imagery, comparable in spatial resolution to aerial mapping imagery, opportunities exist to exploit the inherent spectral information of multispectral imagery to aid urban scene analysis for cartographic feature extraction and simulation database population.

Our previous work has shown the feasibility of merging surface material information derived from moderate resolution multispectral imagery with estimates of height based upon stereo matching in high resolution panchromatic imagery [Ford and McKeown, 1992]. The goal is to use surface material information, normally highly correlated with object location in complex urban scenes, as a source of information for small-scale mapping of man-made structures such as buildings and roads, as well as natural features such as soil, vegetation, and water. The fusion of height estimates with surface material estimates provides a unique synthetic 3-D dataset that is not directly available in any airborne imaging sensor.

Current research involves the generation of surface material classmaps composed of highly confident pixel assignments that will be used in the multispectral fusion process. This procedure of generating high confidence classmaps is described in the following sections.

### 5.1 High Confidence Classmap Generation

In the fusion of surface material and height information, it is important that both datasets be as accurate as possible. In order to generate highly confident surface material classmaps, we have been exploring the use of information available during a Gaussian Maximum Likelihood classification [Foody *et al.*, 1992]. The procedure, outlined in Figure 37, uses a Gaussian Maximum Likelihood classifier with a threshold limit to generate a surface material classmap and a set of discriminant images for each of the surface material types.

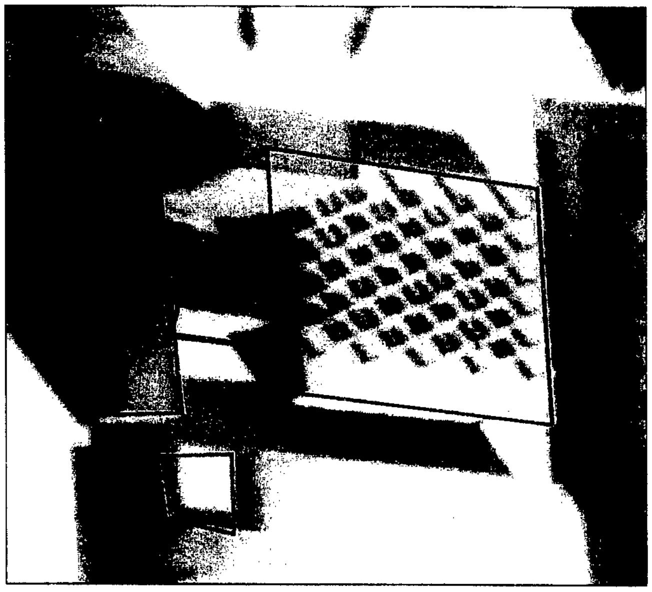


**Figure 35. Surfaces from 3 views**

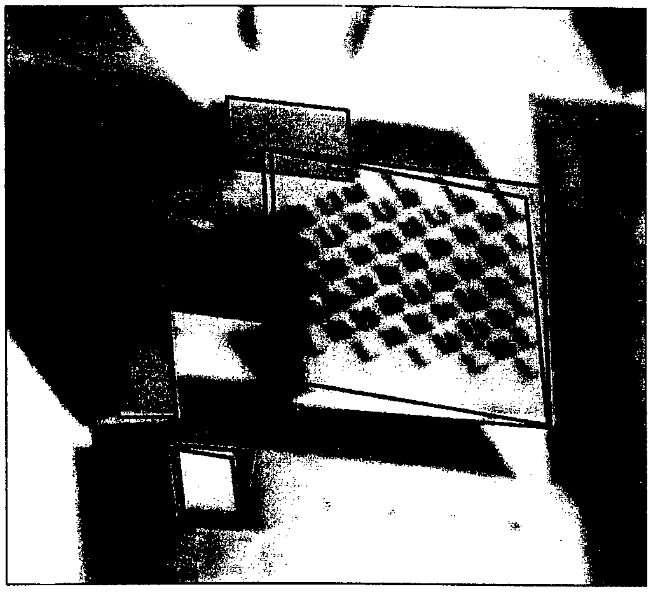


**Figure 36. Surfaces from 5 views**

Using the discriminant images, a discriminant ratio classmap is derived and combined with the Gaussian Maximum Likelihood generated classmap to produce a high confidence surface material classmap.

The following sections describe the threshold and discriminant ratio surface material classmap generation and the merging of these classmaps to construct a high confidence surface material classmap. To illustrate the process, the urban test site, CIVIL1, will be used from the Daedalus ATM multispectral image dataset over Washington, D.C. [Ford and McKeown, 1992]. Figure 38 shows the CIVIL1 urban test site. The scene content is representative of a complex urban area with buildings, street networks and landscaped areas. In order to evaluate the process, a highly detailed surface material ground truth of the CIVIL1 site will be used for accuracy assessment [Ford *et al.*, 1993]. The accuracy assessment is performed at a *fine* and *coarse* surface material level. Table 3 lists the fine surface material members into the five coarse surface material groupings.

### 5.1.1 Threshold Classmap Generation

The Gaussian Maximum Likelihood classification algorithm is one of the most commonly used techniques used in remote sensing applications for generating landcover classmaps from multispectral data; however the effectiveness of this technique is influenced by the training data used. Misclassification can often arise from overlooked or poorly represented classes in the training set. Traditionally this has been handled by applying a threshold to the discriminant function, accepting the $n\%$ most likely pixels from each spectral class [Richards, 1986].

The introduction of a threshold constraint that significantly increases the accuracy of a classification can reject a large percentage of the image pixels. This can be seen in Figure 39, which was generated using a 99 percent threshold Gaussian Maximum Likelihood classification with all 11 Daedalus spectral bands over CIVIL1. Vegetated and shadow areas along with asphalt comprise the majority of the surface materials present in the threshold classmap. Very few concrete features are assigned. Table 4 lists coverage and accuracy of the threshold Gaussian Maximum Likelihood classification against a forced Gaussian Maximum Likelihood classification.

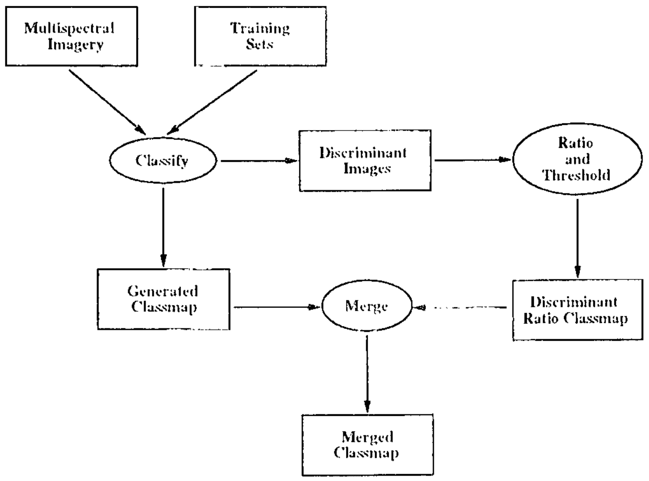


Figure 37. High Confidence Classmap Generation

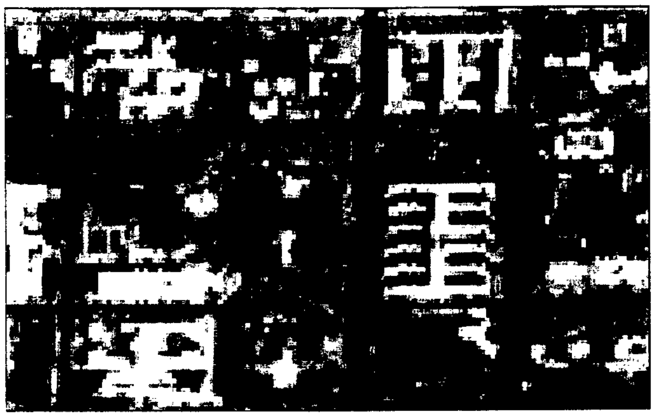


Figure 38. Office of Personnel Management (CIVIL1)

Table 3. Surface Materials

| Fine | Coarse |
|---|---|
| Asphalt<br>Concrete<br>Tile | Man-Made |
| Grass<br>Coniferous Tree<br>Deciduous Tree | Vegetation |
| Shadow | Shadow |
| Soil | Soil |
| Deep Water<br>Shallow Water<br>Turbid Water | Water |

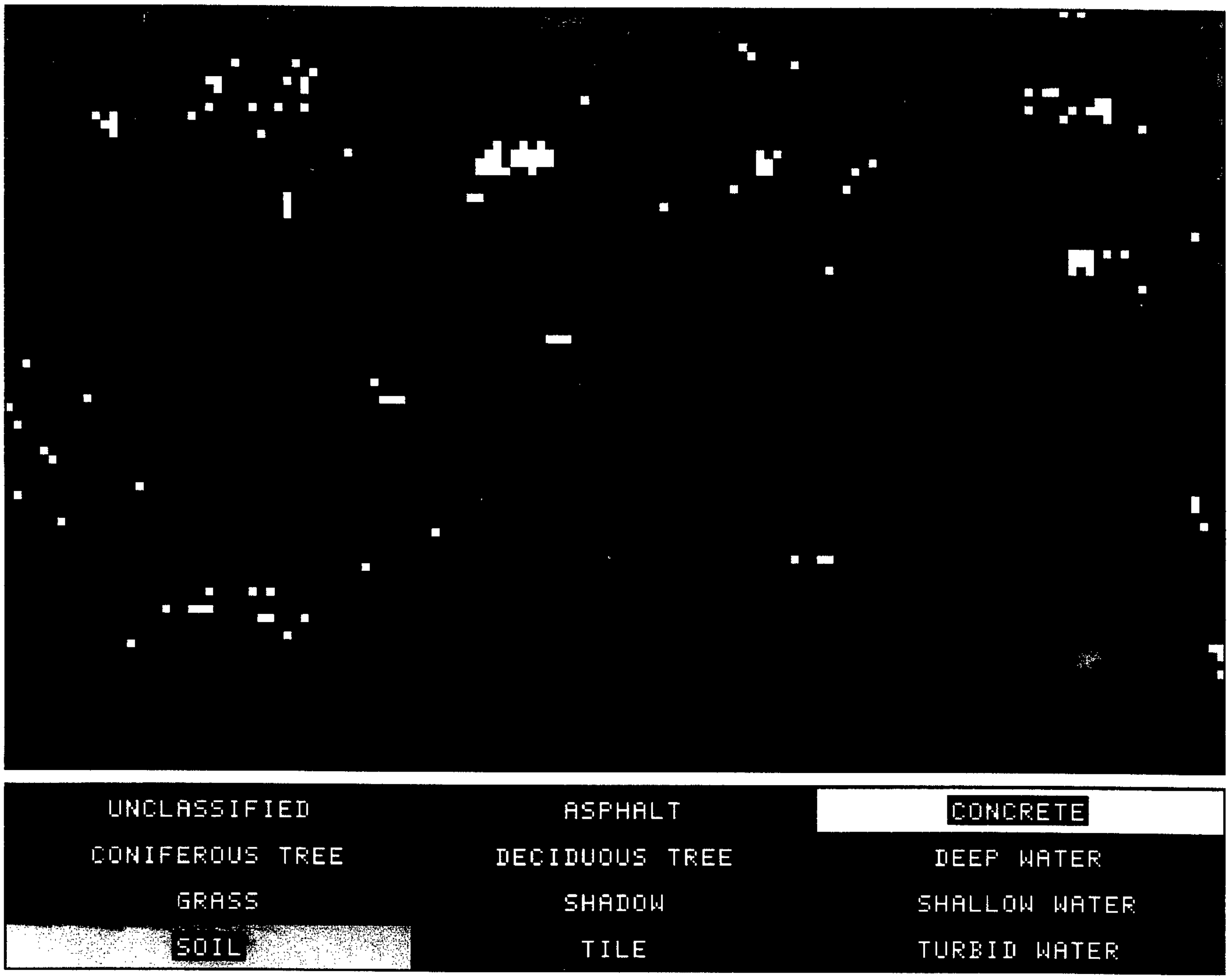


Figure 39. Threshold Classmap

The high accuracy of this result illustrates how the application of thresholds effectively produce classification areas of high confidence, but the coverage of the classification is sparse. The possibility exists for an alternative measure of confidence which could produce highly confident regions to complement those created using the standard threshold. An example of such an alternative measure makes use of discriminant information available from a Gaussian Maximum Likelihood classification as discussed in Section 5.1.2.

### 5.1.2 Discriminant Ratio Classmap Generation

For a given pixel location in a multispectral dataset, the Gaussian Maximum Likelihood classifier calculates a discriminant value for each candidate class based on the class statistics from the training data. A decision rule is then applied to these discriminant values to determine the most likely of the training classes to which the pixel belongs. At this stage, the remaining discriminant values are typically discarded.

One possible measure of confidence in a class assignment exploits the remaining discriminant value possibilities. Using the discriminant values produced by the Gaussian Maximum Likelihood's discriminant function, a ratio between the discriminant values from the second and first most likely classes can be produced. This ratio essentially describes how 'close' the second best class is to the assigned class. Using this ratio, high confidence can be assigned to class assignments where the second and first best classification choices had a great amount of separation. Conversely, low confidence can be assigned to class assignments where the second choice assignment is nearly as likely as the actual assignment.

To test the usefulness of this measure, a Gaussian Maximum Likelihood classification was performed over CIVIL1. During the classification, the discriminant value for each surface material was saved at every pixel location and stored in a set of discriminant images. A ratio between the second and first most likely class assignments was calculated. Figure 40 shows the distribution of these discriminant ratio values for CIVIL1. Larger discriminant ratio values favor higher confidence in the selection of the actual class assignment.

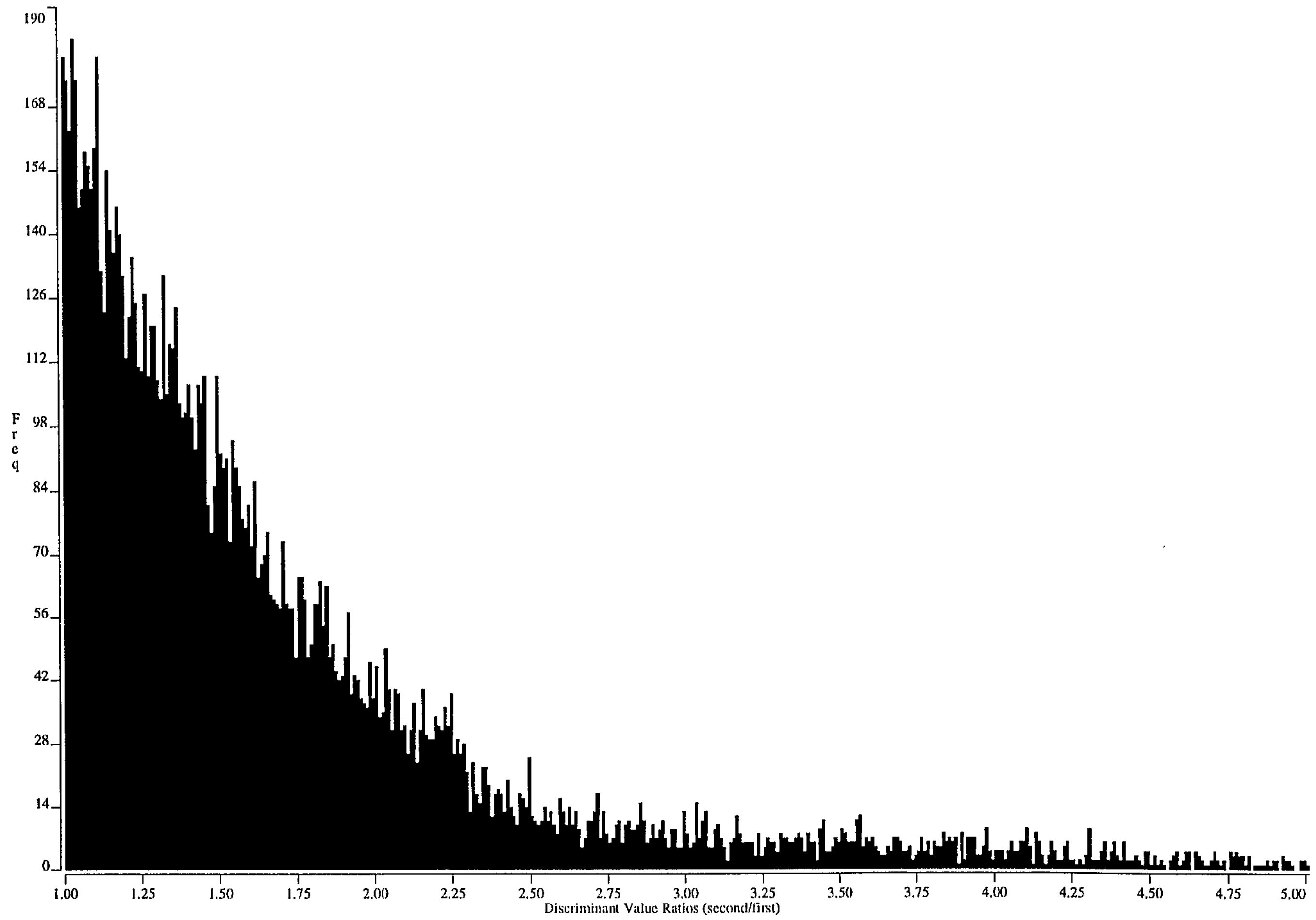
Once the discriminant value ratios are generated for each pixel location, class assignments can be accepted or rejected based on this notion of confidence. Since larger discriminant ratio values favor higher confidence in class selection, Figure 40 suggests that rejecting areas where the discriminant value ratio is 2:1, or lower, should eliminate a large number of misclassified pixels. Applying this ratio threshold means that only those locations where the assigned class is at least twice as likely as the second possibility are accepted into the classification.

The result of applying a 2:1 discriminant ratio value threshold is shown in Figure 41. Vegetated and concrete surface materials mainly compose the discriminant ratio classmap. Unlike the threshold classmap, asphalt is sparsely represented in the discriminant ratio classmap. The accuracy and coverage of the discriminant ratio classmap can be seen in the first column of Table 5.

### 5.1.3 Merging of High Confidence Classmaps

With these high confidence classifications, it is interesting to examine what areas these different methods accept. Figure 41 illustrates areas of high confidence produced using a discriminant value cutoff of 2:1 (i.e. including class assignments where the ratio is 2:1 or higher). Compared to the high confidence areas in Figure 39, which were produced using a threshold, it can be seen that these two methods produce different sets of high confidence pixels.

Figure 42 illustrates a merging of these complementary results, rejecting pixels where the the two classmaps disagree. The merged classmap contains vegetated and shadow areas along with asphalt and concrete features that are an improved representation over the threshold and discriminant ratio classmaps



Civil1 Discriminant Value Ratios

**Figure 40. Discriminant Ratio Histogram**

**Table 4. Threshold Classmap Results**

| | 99% Threshold GML | Forced GML |
|---|---|---|
| Coverage | 24.9% | 100.0% |
| Fine Accuracy | 83.6% | 55.3% |
| Coarse Accuracy | 97.4% | 83.2% |

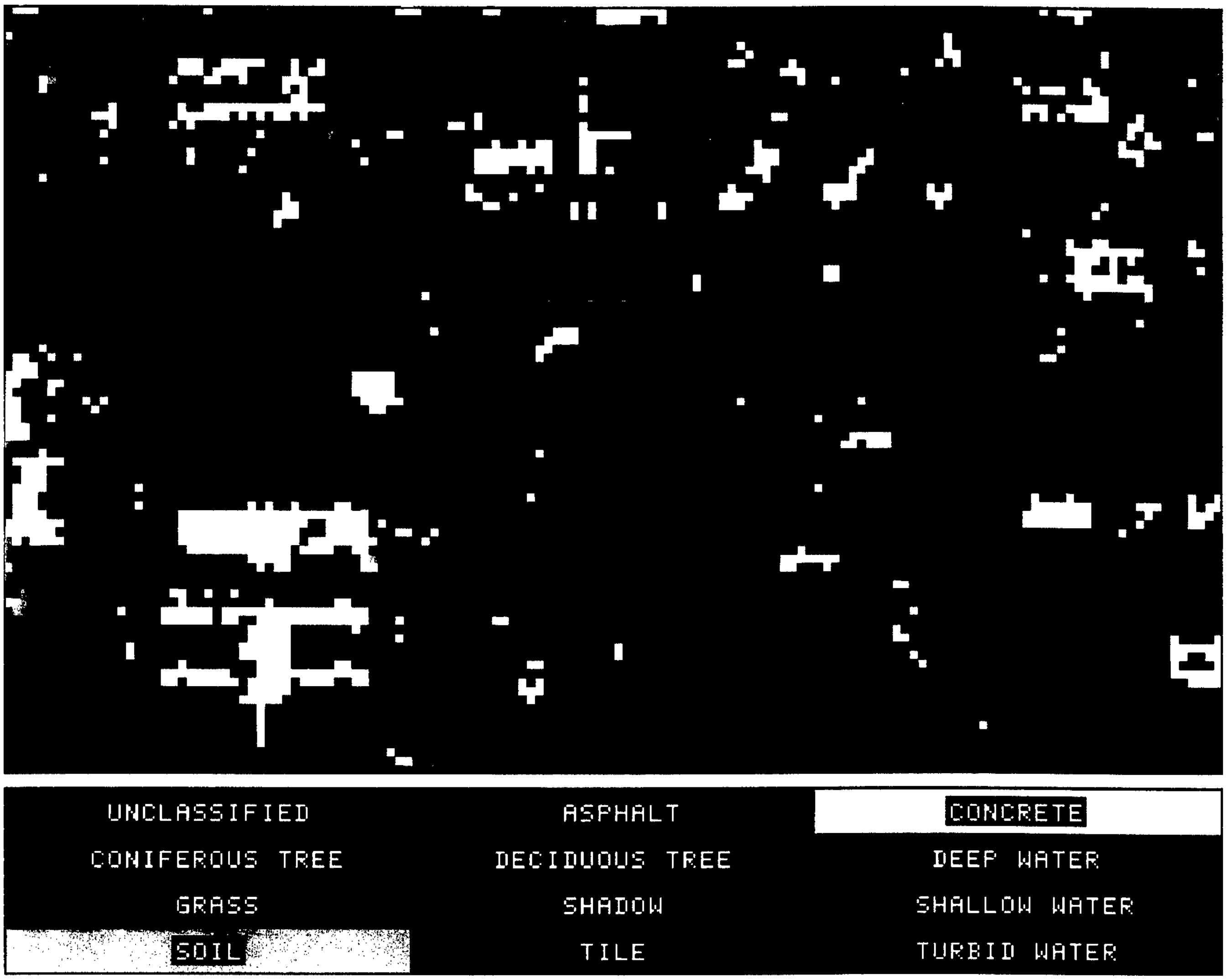


Figure 41. Discriminant Ratio Classmap

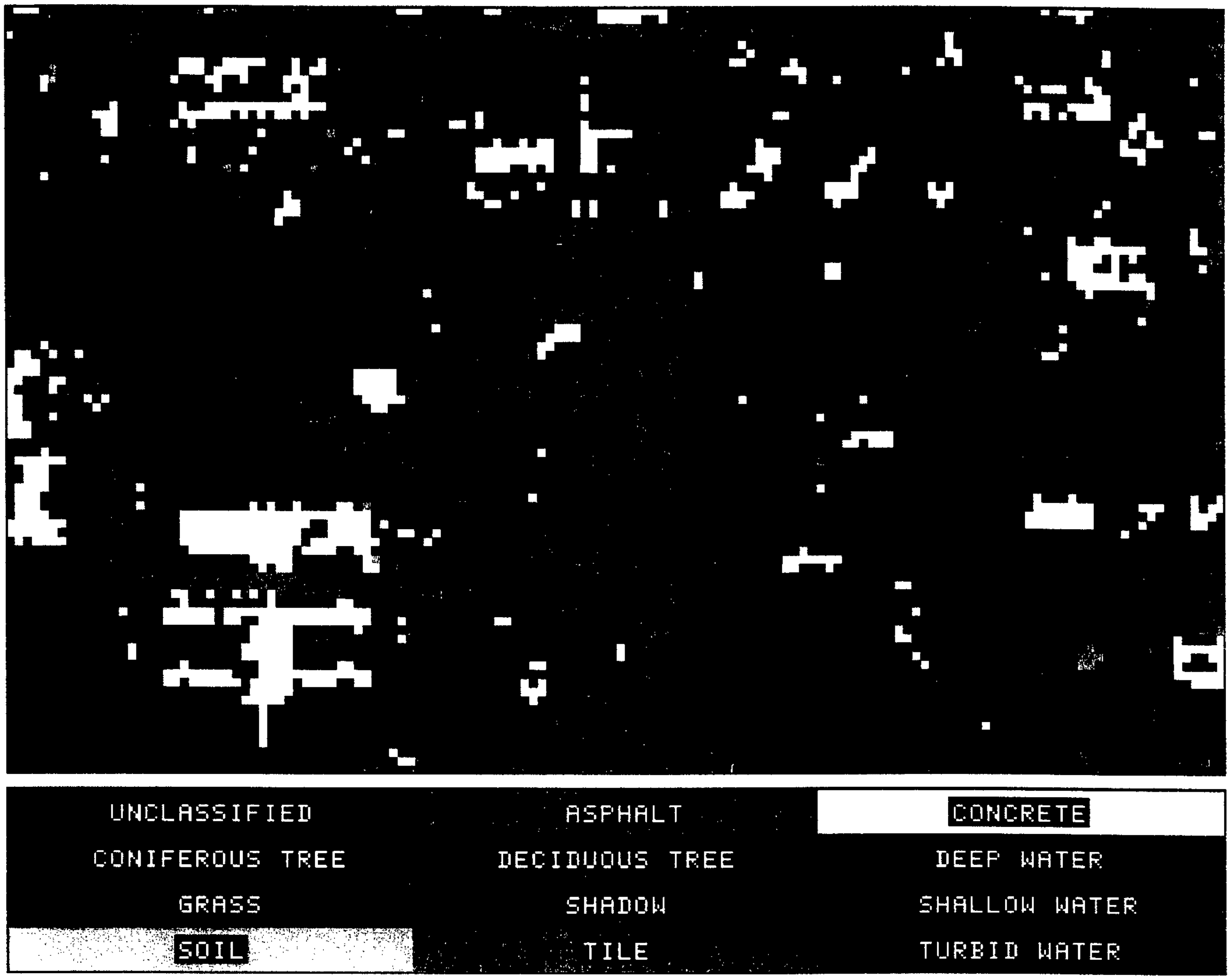


**Figure 42. Merged Classmap**

individually. Significant areas of unclassified pixels are present, resolving their surface material assignment remains as an area of research. Table 6 summarizes the coverage and accuracy results of the merged classmap. The merged classmap has a greater amount of coverage than the individual high confidence classmaps while maintaining a high degree of accuracy. These initial results are promising and support the use of discriminant ratios in generating high confidence areas to complement those produced using a threshold limit.

## 6 Scaling Knowledge-Based Systems

One long term thread of our research in the automated analysis of remotely sensed imagery is the development and use of knowledge-based systems to apply high level structural and spatial constraints for scene interpretation [McKeown *et al.*, 1985; McKeown *et al.*, 1989; Harvey and Tambe, 1993]. More recent work has focused on improving the scalability of large knowledge-based systems for image analysis. In this section we report on two aspects of this work, the use of a portable distributed shared memory system

Table 5. **Discriminant Ratio Classmap Results**

| | 2:1 Discriminant Ratio | Forced GML |
|---|---|---|
| Coverage | 22.9% | 100.0% |
| Fine Accuracy | 73.1% | 55.3% |
| Coarse Accuracy | 98.0% | 83.2% |

Table 6. **Merged Classmap Results**

| | Merged | Forced GML |
|---|---|---|
| Coverage | 37.1% | 100.0% |
| Fine Accuracy | 78.2% | 55.3% |
| Coarse Accuracy | 97.2% | 83.2% |

(MIDWAY) for task-level parallelism and the development of alternative computational models to support explicit parallelism. Both areas are central for the development of analysis systems that use large amounts of image data combined with spatial and structural knowledge.

SPAM was one of the earliest rule-based systems developed specifically for image analysis. SPAMś knowledge is encoded in more than 700 OPS5 production rules. As we continue to improve the interpretation accuracy and SPAMś task domain we continually increase the size of the knowledge base. As the knowledge base and data sets increase in size, additional memory and CPU time is required. We have addressed SPAMś runtime performance, making considerable improvements by using better hardware and software technologies, *e.g.*, converting from Lisp to C, using highly optimized production system languages like CParaOPS5, and running on high-performance workstations. However, in spite of these improvements, the SPAM system can still take four CPU-hours or more to run. These sources of speed-up are insufficient to keep pace with the modifications necessary to expand SPAMś task domain. We have, therefore, continued to pursue work in parallel architectures for production systems, and in language support, for building large rule-based systems.

## 6.1 Task-Level Parallelism Under MIDWAY

Our previous work in parallel architectures developed the concept of task-level parallelism. Task-level parallelism is obtained by decomposing the production system at a high level, thereby exploiting the inherent parallelism in the task. Previously, we investigated the use of task-level parallelism (TLP) [Harvey *et al.*, 1989; Harvey *et al.*, 1990] for speeding up large-scale production systems. SPAM was used as the vehicle for this research performed in conjunction with the Production System Machine group at Carnegie Mellon University. Using task-level parallelism we achieved almost linear speed-ups running on a 16-processor Encore Multimax. In addition, this work produced CParaOPS5 [Acharya and Kalp, 1990], an optimized, C-based version of ParaOPS5 [Kalp *et al.*, 1988] (an assembly language compiler for parallelizing OPS5 programs).

With the availability of high performance uniprocessor workstations now exceeding 100 MIPS, it seemed prudent to suspend our parallelism research until the performance of stable multiprocessor platforms again surpassed uniprocessors. Recently, the MACH group at Carnegie Mellon finished work on an initial implementation of a parallel programming environment called MIDWAY. This environment allows a user to program distributed memory multiprocessors as if they were a shared memory multiprocessor. This is similar to our previous work with the virtual shared memory system [Forin *et al.*, 1989; Li, 1988; Harvey *et al.*, 1990]. In our case, the distributed shared-memory multiprocessor is a set of workstations with a high-speed ATM network interconnect. The MIDWAY system allows us to exploit the inherent parallelism in SPAM while continuing to use the fastest uniprocessor workstations as they become available.

### 6.1.1 MIDWAY

MIDWAY is a system for writing and running shared memory parallel programs on distributed memory multiprocessors [Bershad and Zekauskas, 1991; Bershad *et al.*, 1993]. It combines the programmability of a shared memory multiprocessor with the scalability of a distributed memory system. MIDWAY programs are written in conventional programming languages, such as C and C++. Concurrency within a program is expressed using *threads*, and controlled using *locks* and *barriers*. For example, a MIDWAY program could be written and debugged on a Sequent Symmetry, a true shared memory multiprocessor, and then compiled and relinked for execution on a Delta Touchstone distributed memory multiprocessor.

MIDWAY provides the programmer with a new model of memory consistency called *entry consistency*. Entry consistency takes advantage of the relationship between specific synchronization variables that protect critical sections and the shared data accessed within those critical sections. In an entry consistent system, a processor's view of memory becomes consistent only when it enters a critical section. The only shared memory that is guaranteed to become consistent is that which can be accessed within the critical section. This model provides no greater consistency than that required by most shared memory parallel programs. Consequently, it has the potential for a higher performance implementation than a memory model that delivers "more" consistency than necessary to a program.

MIDWAY consists of three main components: a set of simple extensions to a base language (in this case, C) in the form of keywords and function calls that are used to annotate a parallel program, a runtime system that implements the entry consistency model, and a pre-compiler that translates annotated MIDWAY programs into the underlying base language. The pre-compiler is written in C (actually, extensions have been added to the GNU C compiler). The runtime library provides a portable, machine-independent interface to entry consistent shared memory across a wide range of processor and interconnect architectures. The initial MIDWAY implementation runs on a network of MIPS-based workstations connected by ethernet or ATM/fiber links. Subsequent ports will handle other specialized distributed memory multiprocessors.

### 6.1.2 Implementing Task-Level Parallelism in the SPAM/MIDWAY System

In planning the implementation of the SPAM/TLP system under MIDWAY, we felt it best to begin by attempting to reimplement the task queue processing model we had used in our previous work. This decision has allowed us to use much of what we had previously learned about virtual shared memory, as well as the details of the specific system implementation. Additionally, because we already spent a great deal of time wrestling with the issues of consistency within the SPAM/TLP framework, we felt better prepared to understand the MIDWAY consistency model.

We first evaluated the original SPAM/TLP system from the perspective of mapping data structures to local and/or shared memory locations, locking, synchronization, and minimizing I/O. From this evaluation, we developed the following design outline:

- Incorporate MIDWAY into SPAM

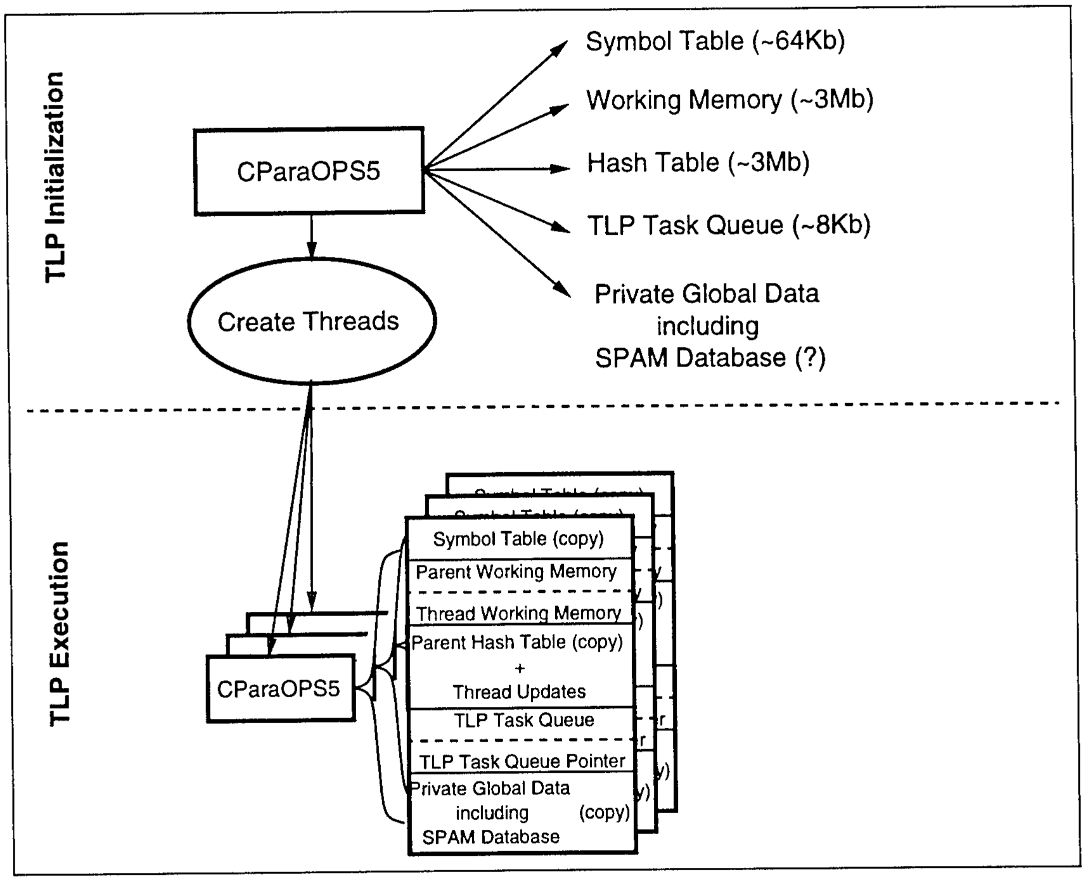


**Figure 43. Organization of the SPAM/MIDWAY system**

- Integrate the TLP task queue model into CParaOPS5
- Incorporate MIDWAY into CParaOPS5.

In our design and implementation of the new SPAM/MIDWAY system, the high-level portion of the SPAM/TLP system did not change appreciably. The SPAM architecture, written in OPS5, was ported to the new system unmodified. The SPAM support library, written in C, required several small changes, primarily to support the change from ParaOPS5 to CParaOPS5. The largest changes were made within the CParaOPS5 implementation to incorporate MIDWAY.

Porting the SPAM/TLP system to CParaOPS5 was primarily a software engineering exercise. CParaOPS5 is a descendent of ParaOPS5, though CParaOPS5 had undergone a number of improvements, making the port more work than just merging old and new versions of software. Once the job of porting the task queue implementation was complete, further changes to CParaOPS5 were required to prepare it to use the MIDWAY system. CParaOPS5 did its own shared memory management, implemented its own locks, and used `fork()` to create child processes. The MIDWAY system provides its own shared memory management scheme, and uses the MACH C-Threads library for locking and thread creation.

With the TLP system integrated into CParaOPS5, we focussed on incorporating MIDWAY into the system. The first task was to carefully lay out our data structures so that data was properly communicated to/from the child processes. As in our previous system, we have two separate processing stages: initialization and task execution. These are shown schematically in Figure 43 separated by a dashed line. During SPAM/TLP initialization phase, CParaOPS5 fires productions to create the TLP task queue. The major OPS5 data structures (the symbol table, working memory, and hash table) are created during initialization. These data are used read-only during the processing by each of the child threads. We therefore allocate these data structures in shared memory and protect each data type with a MIDWAY lock. When the threads acquire a lock, the corresponding data are shipped across the network to the remote machine executing that thread.

Each thread runs on a separate machine. They operate as in the original TLP system, executing as independent OPS5 systems. Also, as before, each thread creates local versions of the working memory and hash table data structures (which grow dynamically during task execution). The local working memory is protected by a MIDWAY lock so that the parent process can gain read access to the data at the end of execution to collect the results. The symbol table and the parent's working memory remain unchanged during task processing; both are available to each of the executing threads.

The TLP task queue created at initialization remains unchanged during task processing and is accessed as read-only shared memory by the threads. A task queue pointer, which keeps track of the current task, is allocated in shared memory and protected by a read-write lock. As in our earlier ParaOPS5/TLP implementation, this is the only shared data structure contended for during task processing. Thus, we expect to see good speedups when we measure performance on the task processing portion of the system.

One final issue for our new system will be the management of shared memory. In our earlier systems — ParaOPS5, ParaOPS5/TLP and CParaOPS5 — we managed the shared memory ourselves by doing block allocation and using type-specific free lists. We have abandoned this scheme and now use MIDWAY system calls to allocate shared data.

The implementation described above provides the same simple task queue processing model that we tested using the ParaOPS5/TLP system. Several idealizations have been made, such as paging over the parent's entire working memory at child startup (instead of paging over portions on demand), or not using the parent process for task execution. These simplifications are useful for accurately understanding the system during this initial implementation phase.

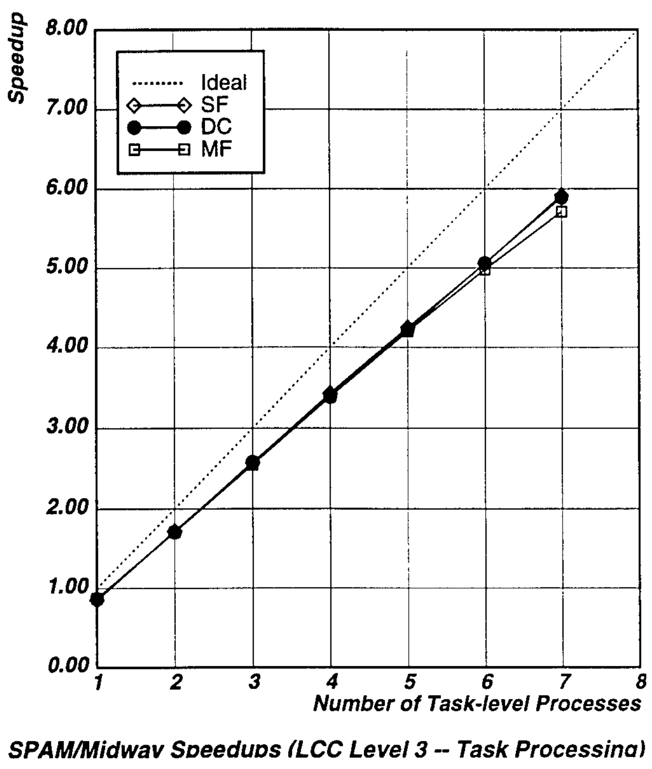
### 6.1.3 SPAM/MIDWAY Results

This initial implementation of SPAM/MIDWAY has been successfully tested on three airport data sets: Moffett Air Force Base (*MF*), Washington National Airport (*DC*) and San Francisco International Airport (*SF*). The speed-up results are computed against a uniprocessor baseline system (called *faketlp*) that has been modified to include the overheads of building and accessing the task queue.

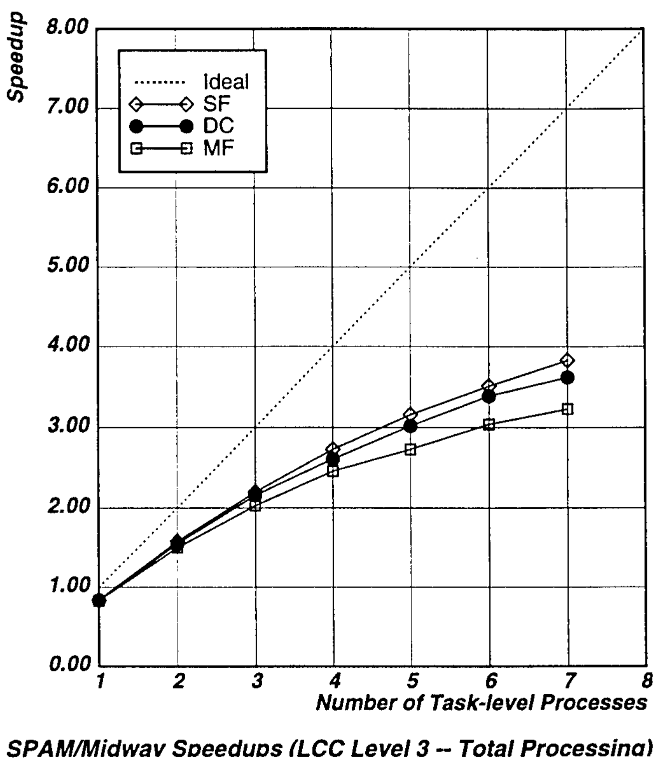
The hardware for our multi-thread experiments consists of eight DEC 5000/200 machines (each rated at approximately 25 MIPS), each with 64 megabytes of memory. All the machines run MACH 3.0 and are linked via fiber-optic cable using Fore Systems' ATM switch.

Our current speed-up results for all three data sets using task-level parallelism under MIDWAY are summarized in Figure 44. Table 7 presents the numbers in more detail. Each row of the table represents a separate system execution. The column labeled Number of Threads is the number of child processes used for that run. The Task Time column presents the amount of time the system spent executing tasks. This begins after the task queue has been set up, and ends before the results are collected together. Although somewhat artificial, this is the accepted method for measuring speed-ups in parallel systems. The Total Time column includes task processing time plus the initialization and result collection times.

We can make several observations from this data. First, the task speed-ups for this system are reasonably good (5.7-fold using 7 child threads) even though we have not tuned the system. The graph in



(a) Task Times



(b) Total Times

Figure 44. Speed-ups varying the number of task-level processes using the MIDWAY virtual shared-memory system with the *MF*, *DC*, and *SF* data sets

Table 7. Speed-ups varying the number of task-level processes using the MIDWAY virtual shared-memory system for the *MF* data set

| Number of Threads | Task Time (sec) | Speed-Up | Total Time (sec) | Speed-Up |
|---|---|---|---|---|
| faketlp | 835.5 | — | 880.7 | — |
| 1 | 985.8 | 0.85 | 1071.5 | 0.82 |
| 2 | 491.1 | 1.70 | 576.9 | 1.53 |
| 3 | 330.0 | 2.53 | 421.3 | 2.09 |
| 4 | 248.9 | 3.36 | 346.5 | 2.54 |
| 5 | 200.0 | 4.18 | 292.3 | 3.01 |
| 6 | 168.1 | 4.97 | 260.9 | 3.38 |
| 7 | 145.6 | 5.74 | 241.1 | 3.65 |

Figure 44 shows that our speed-ups, though below where we would like them to be, again look very linear. If we look at the Task Time data, we can see that the overheads associated with MIDWAY in our system are about 15 percent per processor. The designers report that, in general, the overheads associated with using MIDWAY are about 10–20 percent depending on the amount of communication between threads. These numbers compare favorably. However, when significant amounts of communication occur (such as during the initialization and return-results portions measured as part of the Total Time), the overheads seem somewhat high and are a current topic of investigation.

Although the speed-ups associated with the overall runtime seem low, it should be noted that both the initialization and result collection portions of the execution are currently serialized. These can be parallelized (they were in versions of our network-shared memory system) to improve the overall runtime.

### 6.1.4 Current Status

Though our speed-ups for task processing are good, there is still much to be done. First, we must further reduce the MIDWAY overheads. We are working closely with the designers of MIDWAY to address this problem. Improvements to MIDWAY's internal representations and algorithms are being worked on to reduce these overheads to about 10 percent. Second, more experiments must be done to better understand the system. We are trying other data sets, different hardware configurations, and parallelization of other SPAM phases. All of these will aid in refining and improving the system's performance. Finally, some of the idealizations must be removed from our current system to make it more efficient. As part of this, we must consider the issue of overall runtime if we are to make the system useful for supporting SPAM research.

## 6.2 Language Support for Large Production Systems

In addition to SPAM, several current research efforts are trying to use production system programs for pattern-directed processing on large amounts of data. Examples include maintaining consistency in large databases and triggering actions for unusual and deviant conditions [Schreier *et al.*, 1991], expert databases [Bein *et al.*, 1987; Stonebraker and Hearst, 1988], and simulation [McBride and Lynn, 1990]. A common feature in these applications is that they perform a large number of independent operations that could be performed in parallel. Our preliminary TLP work showed that large amounts of parallelism (50–100 fold) are available in SPAM [Harvey *et al.*, 1991]. This, coupled with the amount of work required to implement TLP in CParaOPS5, has prompted us to investigate the use of different computational models to support large amounts of data and explicit parallelism.

In this section, we describe recent research in the development of an explicitly parallel production language, PPL. PPL has been designed to show that production language programs that process large amounts of data usually contain a large amount of inherent parallelism, and that it is possible to extract this parallelism efficiently by using a language that permits explicit expression of parallelism. We also discuss a new computational model for production systems called *collection-oriented match*, in which conditions, variables, and actions operate on and represent collections of objects. This new matching paradigm is applicable both to new languages, such as PPL, and well as older parallel production languages, such as CParaOPS5.

### 6.2.1 PPL: A Parallel Production Language

Most research into parallel implementations of production system programs has been focused on developing parallelizing compilers for contemporary languages. These languages impose sequentializing constraints

```
(pset not-gate-simulator:on
  (parp not-gate-on
    (not-gate ^input <in> ^output <out>)
    (line ^name <in> ^value on ^time <t>)
  { (line ^name <out>) <output> }
  -->
    (modify <output> ^value off
            ^time (compute <t> + 1))))
```

```
(pset not-gate-simulator:off
  (parp not-gate-off
    (not-gate ^input <in> ^output <out>)
    (line ^name <in> ^value off ^time <t>)
  { (line ^name <out>) <output> }
  -->
    (modify <output> ^value on
            ^time (compute <t> + 1))))
```

**Figure 45. Simulator for circuits consisting of NOT gates**

that preclude the possibility of multiple computations proceeding in parallel. This means that these programs cannot take advantage of parallel algorithms (especially data-parallel algorithms) that could have a large number of independent operations. Parallel Production Language (PPL) [Acharya, 1992] is an explicitly parallel production language that is closely related to the OPS family of languages and is based primarily on OPS5.

For more than one operation to be performed in parallel in a production language program, semantics of the language must allow more than one instantiation to be fired in a single match-resolve-act cycle. Two instantiations firing in parallel can either correspond to the same production or to two different productions. Consider the case when two instantiations of the same production fire in parallel. This will happen when there are multiple similar data items that are independent and can be processed in parallel. On the other hand, parallel firing of two instantiations corresponding to two different productions indicates that there are two different sequences of computation (which may or may not be concerned with the same data) proceeding in parallel.

PPL provides constructs to support both these cases. Situations in which multiple instantiations of the same production fire in parallel can be handled with *parallel productions*. All *active* instantiations[2] of a parallel production fire together. For the purpose of conflict resolution, all instantiations of a parallel production are considered equal. This means that the instantiations of such a production are selected as a single group.

Situations in which instantiations of different productions fire in parallel can be handled with *production-sets*. Production-sets consist of disjoint sets of productions (which may or may not be parallel productions) that fire independent of each other. That means if the conflict set contains instantiations from a production set, at least one of them will be fired irrespective of any instantiations from other production sets that might be present in the conflict set. During conflict resolution, instantiations of all productions in a particular production set are compared only with each other and not with instantiations of productions from other production-sets. A PPL program may contain an arbitrary number of production-sets.

Figure 45 shows an example with two production-sets. Together they implement a fully parallel simulator for circuits consisting of **NOT** gates. Note that all instantiations in the program can be fired in parallel, and the simulation of all operations that happen in one clock tick is done in one match-resolve-act cycle.

The production system computational model is inherently synchronous, *i.e.* matching of all instantiations must be completed before the selection of the dominant instantiations and, once the dominant instantiations are selected, they are fired together. PPL extends the computational model to permit a structured form of asynchronous execution. A PPL program consists of one or more *modules*. Each module

[2]Active instantiations are those that match the current data but have not yet been fired.

**Table 8. Comparison of SPAM implementations in OPS5 and PPL**

**(a) Comparing number of productions per phase.**

| Version | RTF phase | LCC phase | FA phase | Subtotal |
|---|---|---|---|---|
| OPS5 | 209 | 228 | 39 | 476 |
| PPL | 183/225 | 99 | 16 | 340 |

**(b) Comparison of timings for 3 data sets.**

| Version | *MF* data set timing (secs) | *DC* data set timing (secs) | *SF* data set timing (secs) |
|---|---|---|---|
| OPS5 | 1010.1 | 1877.6 | 4470.3 |
| PPL | 571.1 | 1118.1 | 2007.1 |

consists of one or more production-sets and has associated with it a private tuple (working-memory) space and a private conflict set. Instantiations of productions in the same module are fired synchronously whereas instantiations of productions from different modules are fired asynchronously. As discussed previously, instantiations from one production-set are fired in parallel with instantiations from other production-sets in the same module.

PPL extends the standard OPS5 tuple-creation action, `make`, to provide message-passing communication between the different modules that make up a program. Message-passing communication fits in seamlessly with the production-system computational model as individual tuples constitute messages and the different tuple spaces are message destinations.

The computation in a PPL program proceeds in the following fashion:

1. Each module has its own match-resolve-act cycle. At the beginning of the match phase, a module checks if any tuples have been received from other modules. If so, it incorporates these tuples into its tuple space. Thereafter, all productions in the module are matched against contents of the tuple space, the generated instantiations being placed in the conflict set belonging to the module.

2. The resolve phase determines the dominant subset of the conflict set and fires all instantiations that belong to it. The set of operations performed is atomic with respect to the tuple space. Conflict in the act phase could occur if more than one instantiation tries to modify a particular tuple. These conflicts are runtime errors and are flagged as such.

3. If there are no instantiations in the conflict set after the match phase of a cycle, the module waits for a message on the inbound message queue.

4. Different modules execute asynchronously to each other. Synchronization is achieved incorporating the tuples from other modules at the beginning of the match-resolve-act cycle.

Table 8 compares the implementation of SPAM in both OPS5 and PPL. Table 8(a) shows the number of productions used to implement each of SPAM's first three phases. Except for the first phase (RTF), the number of PPL productions used is significantly less than the number of OPS5 productions used. This is mostly because of the elimination of OPS5 productions that are required for managing iteration. PPL easily handles loops because all instantiations of a parallel production are collected and fired together.

The RTF phase is interesting. In the initial implementation, the PPL production count was smaller than the OPS5 count (183 versus 209). However, there were several generic productions associated with RTF evaluation that caused a bottleneck. Because the evaluation productions were generic, SPAM was prevented from doing RTF evaluation until all input objects were looked at. This is not a problem in a serial implementation, but it limits the amount of available parallelism. Restricting these productions to operate on individual object classes solved this problem.

The timings presented in Table 8(b) are presented for all three of the previously mentioned airport data sets (*MF*, *DC* and *SF*). In all cases the PPL implementation ran faster by a factor of 1.8, 1.7, and 2.2, respectively. This speed-up can be attributed to the improved expressiveness of the language (*e.g.*, PPL's implicit loops versus OPS5's explicit loops). as well as to improvements to internal algorithms (*e.g.*, conflict set management).

### 6.2.2 Collection-Oriented Match

Current computational models for production systems are suitable for expert systems dealing with small amounts of data (which they use for tasks such as diagnosis and configuration), or for cognitive modeling. Such models, and languages based on such models, have features that limit scalability. In particular, current match algorithms work well for small working memories. They do not scale well with the number of tuples. The primary reason for this is that they try to match each tuple individually and do not attempt to take advantage of structure that might implicitly be present in the tuples. A large growth in the number of tuples can lead to a combinatorial explosion in the number of tuple combinations that need to tested for match.

An alternate computational model based on collection-oriented semantics has been developed. In this model, conditions match collections of tuples instead of individual tuples, variables are bound to collections of values, and actions operate on collections of values or tuples. A single instantiation in this model can process a collection of arbitrary size. This model tries to take advantage of structure implicit in the tuples present in the working memory. It has four advantages over contemporary models:

- It allows collection-oriented operations on large sets of tuples, *e.g.* mean, average, variance, filtering, etc. Contemporary models do not support these operations.
- It provides a more natural style of operating on collections. There is no longer a need to contort the production system model to support collection-oriented operations. Several phases of SPAM process large collections (*e.g.*, collections of two-dimensional regions, collections of hypotheses, etc.). This should also allow us to simplify and extend RULEGEN, the SPAM knowledge compiler.
- It allows programmers to use efficient collection-oriented libraries developed in other languages. Such libraries have been developed in the data parallel programming community.
- It permits significantly more efficient match algorithms. Tuple-oriented semantics requires that individual tuples be matched separately. Collection-oriented semantics allow collections of tuples to be matched as a unit that could potentially tame the combinatorial explosion in the number of tests as the number of tuples grows.

### 6.2.3 Current Status

An implementation of PPL on shared-memory machines has been developed and is being tested. Current work includes implementation of a large and diverse set of tasks in PPL and analysis of the results. The implementation of PPL is based, in part, on the ParaOPS5 [Kalp *et al.*, 1988] and CParaOPS5 [Acharya and Kalp, 1990] implementations. Portability is an important goal. Like the CParaOPS5 compiler, the PPL compiler generates fully portable C code. The runtime system is mostly portable, with the machine dependencies being limited to code for implementing simple locks.

The collection-oriented match computational model has been developed to the point that it can be used for language design. The preliminary implementation is called Collection-oriented Production Language

(COPL) [Acharya and Tambe, 1992]. It is a variant of OPS5 that uses the collection-oriented semantics described above. Though the implementation is incomplete, several programs have been written in COPL and significant speed-ups have been achieved. Preliminary evaluation has indicated that the scaling characteristics of programs in COPL are significantly better than corresponding programs in OPS5.

## 6.3 Future Research Directions

In the future, we plan to combine much of our work in the areas of task-level parallelism and language support to directly impact the performance, in terms of speed and accuracy, of our rule-based scene interpretation system, SPAM. The major topics for this integration of effort are listed below.

- SPAM and MIDWAY — We are in the final phases of completing an implementation of SPAM/MIDWAY that will use parallelism in three out of the four SPAM phases (RTF, LCC, and FA). Further efficiency considerations within both SPAM/TLP and MIDWAY are being addressed in tandem.
- PPL and MIDWAY — The PPL language is engineered in such a way as to attempt to simplify porting the language to other shared memory systems. The MIDWAY system is an excellent candidate for testing these engineering decisions. PPL's explicit parallelism model will potentially simplify the integration of task-level parallelism into the current (and future) versions of SPAM. Additionally, this would positively impact our ability to make changes to the SPAM/MIDWAY TLP system.
- COPL and PPL — The collection-oriented semantics developed and implemented in COPL do not conflict with the parallel semantics developed and implemented in PPL. An interesting research area would be to combine these semantics into a single language.

## 7 Automatic Construction of Virtual Worlds

There is increasing interest in developing virtual world databases for advanced distributed simulation that maintains a high degree of fidelity with respect to the actual 3-D environment. The compilation of such spatial databases requires the integration of information from a variety of sources, including digital map data, aerial imagery, detailed line drawings, and ground-based photography. Current applications for such databases have revolved around distributed interactive simulation addressing military training, primarily in the areas of mission planning and rehearsal, and in vehicle simulation.

Factors that currently limit the development of such systems include the lack of terrain and environmental data at the level of detail required for ground-based simulations, the magnitude of the scene generation process, and the inadequacy of input devices that allow a user to interact with the virtual world.

Much of the research on automated feature extraction described in the earlier sections of this paper can be applied to the construction or intensification of simulation databases. For example, manual compilation of building models and road networks are particularly time-consuming, and automated, and semi-automated analysis methods can speed up the database intensification process. Our research in multispectral analysis is particularly appropriate for increasing simulation fidelity, in its ability to establish surface material categories for improved visualization of man-made and natural features.

We have been primarily concerned with the generation of virtual world databases, and have not investigated real-time user interaction. The issues we have addressed are the:

- generation of an efficient terrain model suitable for real-time visualization
- integration of road networks with the terrain model to form a coherent terrain skin

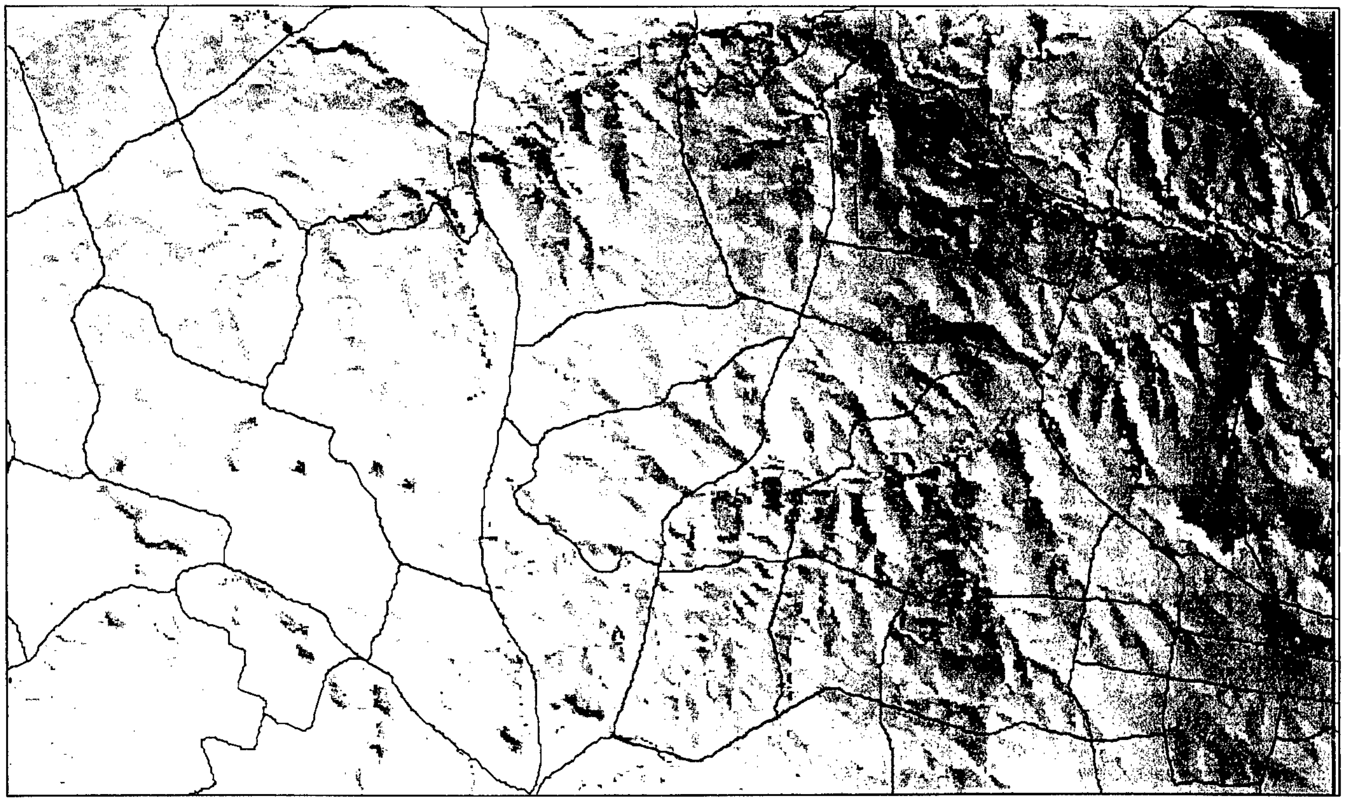


**Figure 46. Ft. Hood database**

- control of polygon complexity in urban areas that might otherwise overload simulation hardware, taking both roads and terrain complexity into account
- integration of cultural features extracted from aerial imagery.

In this section we give a brief overview of our current research issues in database construction for advanced simulation. They are are addressed in greater detail in [Polis *et al.*, 1994].

## 7.1 Terrain Skin Generation

The terrain skin is the substrate on which the entire virtual world is built up. The representation for the virtual world can range from a flat plane to a detailed polygonal mesh. Feature data describing transportation networks, drainage, vegetation, and man-made features, are usually represented as 2-D planimetric features, and their elevation is derived from the underlying terrain. The integration of these features usually requires the rationalization of mismatches in positional accuracy and level of detail. Further, issues of intervisibility that have not been critical in flight simulators, became key factors for ground-based simulations. Representations that present an overly smooth terrain skin will fail to satisfy requirements for realism.

We have found that a triangular irregular network (TIN) takes maximum advantage of existing computer image generator (CIG) hardware. It can represent irregular, as well as regular, polygonal meshes, taking full advantage of the rendering hardware. The method used for generating TINs is described in detail in [Polis and McKeown, 1993]. We iteratively refine the TIN by adding points where there is maximum global error. This method has proved to be very flexible. We have been able to incorporate user-specified importance regions, such as mounts, alluvial fans. and plains, to achieve selective fidelity in the resulting TIN. Our method can avoid terrain refinement in those areas where further detail would make the terrain too complex for CIG display. Finally, we have demonstrated the automated integration of the road network with the terrain to ensure proper trafficability.

## 7.2 Road Integration

There are a few key features in a terrain based virtual world that are important to ground simulators. One of these features is the transportation network represented by roads of various types. Ideally, this road network should be internally consistent, and integrated into the terrain model directly.

Previous methods for road network integration are suited to flat terrain. Roads are usually pasted directly on the terrain skin without modification of the underlying terrain. We have developed a superior approach that integrates roads directly into the terrain model and corrects the terrain accordingly. So, for example, extreme road tilts and grades that often appeared using previous methods can be removed during the integration process.

## 7.3 Using Automatically Extracted Cartographic Features

One key issue in the rapid construction of virtual worlds is the ability to update existing cartographic sources with information extracted from imagery taken in a timely manner. There is a large body of research, primarily derived from the ARPA Image Understanding community, related to cartographic update, site model construction, and analysis of remotely sensed imagery [DAR, 1993]. Our research has focused on the extraction of detailed road networks and man-made structures such as buildings [McKeown and others, 1993]. Examples of the use of derived features are shown in [Polis *et al.*, 1994].

## 7.4 Ft. Hood Database

Figure 46 shows an overview of a virtual world representing part of Ft. Hood, Texas. The database includes a TIN terrain skin, integrated DMA ITD roads, and automatically extracted buildings. Here we show a shaded relief of the underlying terrain with roads superimposed as linear features. This database is of medium size and is of particular interest since it is composed by integrating several widely divergent data sources. The ITD road network for this area is very dense, and contains many road segments per load module. As such, integrating roads into the terrain added about 6 percent to the polygon count over simply pasting the roads onto the terrain skin.

## Acknowledgments

We gratefully acknowledge Anurag Acharya and Dirk Kalp (of the Production System Machine Group) and Wayne Sawdon, Matt Zekauskas, and Brian Bershad (of the MIDWAY Group) for their contributions to the research in scaling of knowledge-based systems.

We also thank the unsung hacker-youth of the Digital Mapping Laboratory for their contributions to various aspects of the research reported in this report: Ed ‘Finesse’ Allard, Michael ‘Armadillo’ Bowling, Ted ‘Rocky’ Bulwinkle, Michael ‘MSG’ Higgins, Christian ‘’ Hoffman, Nathan ‘Atonal’ Segerlind, Mark ‘Flipper’ Stemm, Bliksem ‘Al’ Tobey and Erik ‘Cheesedog’ Witte; as well as Chris ‘IDL’ Olson and Jeff ‘Presto’ McMahill.

This report appeared in the Proceedings of the 1994 ARPA Image Understanding Workshop, 14–18 November 1994, held in Monterey CA.

For more information regarding our research activities see our webpage at:

`http://www.cs.cmu.edu/~MAPSLab/`

The research reported in this report was supported by the Advanced Research Projects Agency (ARPA/SISTO) and the U.S. Army Topographic Engineering Center under Contracts DACA76-91-C-0014 and DACA76-92-C-0036, and the Air Force Office of Scientific Research (AFOSR) under Contract F49620-92-J-0318. Student research and training support was provided by the U.S. Army Research Office ASSERT program under Contract DAAH04-93-G-0092. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Topographic Engineering Center, the U.S. Army Research Office, the Air Force Office of Scientific Research, or the Advanced Research Projects Agency, or of the United States Government.

## References

Acharya and Kalp, 1990. A. Acharya and D. Kalp. Release notes for CParaOPS5 5.3 and ParaOPS5 4.4. This is distributed with the CParaOPS5 release, 1990.

Acharya and Tambe, 1992. A. Acharya and M. Tambe. Collection-oriented match: Scaling up the data in production systems. Technical Report CMU-CS-92-218, School of Computer Science, Carnegie Mellon University, Dec. 1992.

Acharya, 1992. A. Acharya. PPL: An explicitly parallel production language for large scale parallelism. In *Proceedings of the IEEE conference on Tools for AI*, pp. 473–474, 1992.

Aviad, 1988. Z. Aviad. Locating corners in noisy curves by delineating imperfect sequences. Technical Report CMU-CS-88-199, Carnegie Mellon University, Dec. 1988.

Barnard, 1983. S. Barnard. Interpreting perspective images. *Artificial Intelligence*, 21:435–462, 1983.

Bein *et al.*, 1987. J. Bein, R. King and N. Kamel. MOBY: An architecture for distributed expert database systems. In *Proceedings of the Thirteenth International Conference on Very Large Databases*, pp. 13–20, 1987.

Bershad and Zekauskas, 1991. B. Bershad and M. Zekauskas. Midway: Shared memory parallel programming with entry consistency for distributed memory multiprocessors. Technical Report CMU-CS-91-170, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, Apr. 1991.

Bershad *et al.*, 1993. B. Bershad, M. Zekauskas and W. Sawdon. The midway distributed shared memory system. Technical Report CMU-CS-93-119, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, Mar. 1993.

Cochran, 1994a. S. D. Cochran. Adaptive vergence for stereo matching. In *Int. Archives of Photogrammetry and Remote Sensing: Spatial Information from Digital Photogrammetry and Computer Vision*, vol. 30, 3/1, pp. 144–151, Munich, Germany, 5-9 Sept. 1994. Commission III.

Cochran, 1994b. S. D. Cochran. Adaptive vergence for the stereo matching of oblique imagery. In *ARPA IUW*, pp. 1335–1348, Monterey, CA, 13–16 Nov. 1994. ARPA, Morgan Kaufmann.

DAR, 1993. DARPA. *DARPA IUW*, DC, 19–21 Apr. 1993. Morgan Kaufmann Publishers.

Foody *et al.*, 1992. G. M. Foody *et al.* Derivation and applications of probabilistic measures of class membership from the maximum-likelihood classification. *PERS*, 58(9):1335–1341, Sept. 1992.

Ford and McKeown, 1992. S. J. Ford and D. M. McKeown, Jr. Utilization of multispectral imagery for cartographic feature extraction. In *DARPA IUW*, pp. 805–820, San Diego, CA, Jan. 1992. DARPA, Morgan Kaufmann.

Ford *et al.*, 1993. S. J. Ford, J. B. Hampshire and D. M. McKeown, Jr. Performance evaluation of multispectral analysis for surface material classification. In *DARPA IUW*, pp. 421–435, DC, Apr. 1993. DARPA, Morgan Kaufmann.

Forin *et al.*, 1989. A. Forin, J. Barrera and R. Sanzi. The shared memory server. In *Proceedings of USENIX — Winter 89*, pp. 229–243, Jan. 1989.

Fua and Hanson, 1991. P. Fua and A. J. Hanson. An optimization framework for feature extraction. *Machine Vision and Applications*, 4(2):59–87, 1991.

Harvey and Tambe, 1993. W. Harvey and M. Tambe. Experiments in knowledge refinement for a large rule-based system. Technical Report CMU–CS–93–195, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, Aug. 1993.

Harvey *et al.*, 1989. W. Harvey *et al.* Measuring the effectiveness of task-level parallelism for high- level vision. In *DARPA IUW*, pp. 916–933. Morgan Kaufmann, May 1989.

Harvey *et al.*, 1990. W. Harvey *et al.* The effectiveness of task-level parallelism for high-level vision. In *Proceedings of the 2nd ACM/SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, pp. 156–167, Mar. 1990.

Harvey *et al.*, 1991. W. Harvey *et al.* The effectiveness of task-level parallelism for production systems. *J. Parallel and Distributed Comp.*, 13, Dec. 1991.

Huertas and Nevatia, 1988. A. Huertas and R. Nevatia. Detecting buildings in aerial images. *CVGIP*, 41(2):131–152, Feb. 1988.

Irvin and McKeown, 1989. R. B. Irvin and D. M. McKeown. Methods for exploiting the relationship between buildings and their shadows in aerial imagery. *IEEE Trans. SMC*, 19(6):1564–1575, 1989. Also available as Technical Report CMU–CS–88–200, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

Kalp *et al.*, 1988. D. Kalp *et al.* Parallel OPS5 user's manual. Technical Report CMU-CS-88-187, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, Nov. 1988.

Li, 1988. K. Li. IVY: A shared virtual memory system for parallel computing. In *Proceedings of International Conference On Parallel Processing*, pp. 94–101, 1988.

Lin *et al.*, 1994. C. Lin, A. Huertas and R. Nevatia. Detection of buildings using perceptual grouping and shadows. In *CVPR*, pp. 62–69, Seattle, WA, 19–23 June 1994.

Liow and Pavlidis, 1990. Y.-T. Liow and T. Pavlidis. Use of shadows for extracting buildings in aerial images. *CVGIP*, 49(2):242–277, Feb. 1990.

McBride and Lynn, 1990. R. A. McBride and K. Lynn. Anatomy of rule-based simulation. In *Proceedings of the SCS Multiconference 1990*, pp. 24–9, Jan. 1990.

McGlone and Shufelt, 1993. J. C. McGlone and J. A. Shufelt. Incorporating vanishing point geometry into a building extraction system. In *DARPA IUW*, pp. 437–448, DC, 19–21 Apr. 1993. DARPA, Morgan Kaufmann.

McGlone and Shufelt, 1994. J. C. McGlone and J. A. Shufelt. Projective and object space geometry for monocular building extraction. In *CVPR*, pp. 54–61, Seattle, WA, 19–23 June 1994.

McGlone, 1992. J. C. McGlone. Design and implementation of an object-oriented photogrammetric toolkit. In *Int. Archives of Photogrammetry and Remote Sensing*, vol. XXIX, B2, pp. 334–338, 1992.

McKeown and Lukes, 1992. D. M. McKeown, Jr. and G. E. Lukes. Building a cartographic infrastructure: Simulation and cartography. In *DARPA IUW*, pp. 265–268, San Diego, CA, Jan. 1992. DARPA, Morgan Kaufmann.

McKeown and others, 1993. D. M. McKeown *et al.* Research in automated analysis of remotely sensed imagery. In *DARPA IUW*, pp. 231–252, DC, 19–21 Apr. 1993. DARPA, Morgan Kaufmann.

McKeown *et al.*, 1985. D. M. McKeown, W. A. Harvey and J. McDermott. Rule based interpretation of aerial imagery. *IEEE Trans. PAMI*, PAMI-7(5):570–585, Sept. 1985.

McKeown *et al.*, 1989. D. M. McKeown, W. A. Harvey and L. Wixson. Automating knowledge acquisition for aerial image interpretation. *CVGIP*, 46(1):37–81, Apr. 1989.

McKeown, 1990. D. M. McKeown, Jr. *Toward Automatic Cartographic Feature Extraction* in Mapping and Spatial Modeling for Navigation, vol. 65 of *NATO ASI Series F: Computer and Systems Sciences*, pp. 149–180. Springer-Verlag, 1990. Edited by L. Pau.

Mohan and Nevatia, 1989. R. Mohan and R. Nevatia. Using perceptual organization to extract 3-D structures. *IEEE Trans. PAMI*, 11(11):1121–1139, Nov. 1989.

Nevatia and Babu, 1980. R. Nevatia and K. R. Babu. Linear feature extraction and description. *CGIP*, 13:257–269, July 1980.

Nicolin and Gabler, 1987. B. Nicolin and R. Gabler. A knowledge-based system for the analysis of aerial images. *IEEE Trans. GRS*, GE–25(3):317–329, May 1987.

Norvelle, 1992. F. R. Norvelle. Stereo correlation: Window shaping and DEM corrections. *PERS*, 58(1):111–115, Jan. 1992.

Polis and McKeown, 1993. M. F. Polis and D. M. McKeown, Jr. Issues in iterative TIN generation to support large scale simulations. In AUTOCARTO 11*: International Symposium on Computer Assisted Cartography*, pp. 267–277, Minneapolis, MN, 30 Oct.–1 Nov. 1993.

Polis *et al.*, 1994. M. F. Polis, S. J. Gifford and D. M. McKeown, Jr. Automating the construction of large scale virtual worlds. In *ARPA IUW*, pp. 931–946, Monterey, CA, 13–16 Nov. 1994. ARPA, Morgan Kaufmann. Also available as Technical Report CMU–CS–94–199, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

Quam, 1984. L. H. Quam. Hierarchical warp stereo. In *DARPA IUW*, pp. 149–155. DARPA, Oct. 1984.

Richards, 1986. J. A. Richards. *Remote Sensing Digital Image Analysis: An Introduction.* Springer-Verlag, Berlin, 1986.

Roux and McKeown, 1994a. M. Roux and D. M. McKeown, Jr. Feature matching for building extraction from multiple views. In *CVPR*, pp. 46–53, Seattle, WA, 19–23 June 1994.

Roux and McKeown, 1994b. M. Roux and D. M. McKeown, Jr. Feature matching for building extraction from multiple views. In *ARPA IUW*, pp. 331–349, Monterey, CA, 13–16 Nov. 1994. ARPA, Morgan Kaufmann.

Schreier *et al.*, 1991. U. Schreier *et al.* Alert: An architecture for transforming a passive DBMS into an active DBMS. In *Proceedings of the 17th VLDB Conference*, 1991.

Shufelt and McKeown, 1993. J. A. Shufelt and D. M. McKeown. Fusion of monocular cues to detect man-made structures in aerial imagery. *CVGIP*, 57(3):307–330, May 1993.

Stonebraker and Hearst, 1988. M. Stonebraker and M. Hearst. Future trends in expert database systems. In *Proceedings of the Second International Conference on Expert Database Systems*, pp. 3–20, 1988.