TEC-0062

A System for Automated Sight Model Acquisition and Extension - RADIUS

Robert Collins
Allen Hanson
Edward Riseman

University of Massachusetts
Department of Computer Science
P. O. Box 34610
Amherst, MA 01003-4610

June 1995

Prepared for:
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, Virginia 22315-3864

US Army Corps
of Engineers
Topographic
Engineering Center

TEC

Destroy this report when no longer needed.
Do not return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

The citation in this report of trade names of commercially available products does not constitute official endorsement or approval of the use of such products.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
June 1995

3. REPORT TYPE AND DATES COVERED
First Annual Oct. 1992 - Oct. 1993

4. TITLE AND SUBTITLE

A System for Automated Sight Model Acquisition and
Extension - RADIUS

5. FUNDING NUMBERS

DACA76-92-C-0041

6. AUTHOR(S)

Robert Collins Edward Riseman
Allen Hanson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Massachusetts
Department of Computer Science
P. O. Box 34610
Amherst, MA 01003-4610

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Advanced Research Projects Agency
3701 North Fairfax Drive, Arlington, VA 22203-1714

U.S. Army Topographic Engineering Center
7701 Telegraph Road., Alexandria, VA 22315-3864

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

TEC-0062

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The goal of the Research and Development for Image Understanding Systems (RADIUS) project is to develop image understanding (IU) algorithms that support model-based aerial photo interpretation. This report, funded under a three-year contract to develop algorithms for automated geometric site modeling, describes research activities performed during the first year, October 1992 through October 1993.

14. SUBJECT TERMS

Aerial image understanding, feature extraction, site modeling,
image extraction

15. NUMBER OF PAGES
54

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

Contents

List of Figures

List of Tables

PREFACE

This research is sponsored by the Advanced Research Projects Agency (ARPA) and monitored by the U.S. Army Topographic Engineering Center (TEC) under Contract DACA76-92-C-0041, titled "The UMass RADIUS Project - Year 1". The ARPA Program Manager is Dr. Oscar Firschein, and the TEC Contracting Officer's Representative is Ms. Lauretta Williams.

1 Introduction

The goal of the Research and Development for Image Understanding Systems (RADIUS) project is to develop image understanding (IU) algorithms that support model-based aerial photointerpretation. The Computer Vision Research Laboratory at the University of Massachusetts (UMass) is funded under a three-year contract to develop algorithms for automated geometric site modeling. Delivery will be in the form of a set of IU modules, each containing a flexible set of programs for performing modeling tasks. Although each individual task is highly automated, the image analyst (IA) maintains control of the site modeling session by deciding when and where each program is applied.

Our goal is to provide automated IU support for the site modeling process. Given a current partial site model, and a set of images of the site, we provide IU routines that extend the model to include previously unmodeled site features (**model extension**), and that reduce the inaccuracies in the existing model (**model refinement**). This process is repeated as new images become available, each updated model becoming the current site model for the next iteration. Over time, the site model will be steadily improved to become more complete and more accurate. Routines for **model-to-image registration** support these tasks by automatically determining the relative position and orientation of each image with respect to the local site coordinate system. This information is needed for projecting the current model onto the new image for change detection, and to determine what parts of the new image are not currently covered in the site model. The most difficult modeling task to automate is **model acquisition**, the generation of an initial site model from scratch. Early versions of the system will rely on IA guidance to perform this important task, although work is underway to more fully automate this process as well.

This report describes research activities performed during the first year of the UMass RADIUS contract, October 1992 – October 1993. The rest of this section presents a brief overview of the project; subsequent sections are devoted to in-depth technical discussions of the algorithms that have been developed so far.

1.1 Program Activities

UMass plans to deliver a set of software modules that provide IU support for site model acquisition, extension, and refinement. Each module is listed below, along with a brief description.

1. The **Feature Extraction** module generates symbolic feature descriptions of incoming imagery in order to reduce the representational gap between incoming sensor data and stored geometric site model descriptions. Algorithms are currently available for *straight line segment extraction* and *corner detection.*

2. The **Model-to-Image Registration** module registers incoming imagery with a store geometric site model by aligning their respective coordinate systems. This process allows a site model graphic to be overlayed on the image and is the first step in model extension and refinement. Registration is achieved in two stages: *3D-to-2D feature correspondence match-*

ing, which determines the correspondence between site model features and extracted image features, and *robust pose determination;* which uses these correspondences to determine the precise position and orientation of the camera in the scene.

3. The **Model Extension** module contains routines used for site model acquisition, extension and refinement. Given a set of interesting features seen in one image, an *epipolar feature matching* procedure uses known image poses to search for corresponding geometric features in other images taken from different viewpoints. Corresponding features are passed to a *multi-image triangulation* routine that determines the position of the new 3D features in the local site coordinate system. The triangulation routine can also be used to refine the positions of old model features based on new observations detected via model-to-image registration. This module is not yet fully developed, and may eventually contain a *building detection* routine for automatically finding new buildings to be added to the site model.

4. The **Vanishing Point Analysis** module can be used to detect groups of oriented man-made structures in the scene, to determine the relative orientation of 3D line structures from a single image, and to generate an initial estimate of the rotational component of camera pose. An algorithm for *vanishing point detection* groups extracted straight line segments into sets of lines directed towards the dominant scene vanishing points. Precise statistical estimation of *vanishing point orientation* is provided for determining the relative orientation of lines in the scene with respect to the camera, and vice versa.

5. The **Image-to-Image Registration** module determines feature correspondences directly between two or more images without going through a model-to-image registration step. It is thus useful when no models are yet available, as is the case during initial model acquisition. When the pose of each image is already known, the epipolar feature matching algorithm included in the Model Extension module can be used. When the pose is not known, epipolar matching cannot be used – for this case we have experimented with an algorithm for *automated image rectification* that transforms the unconstrained perspective image-to-image feature matching problem into one that can be solved using simpler similarity matching. In the future, this module will contain a *correlation-based stereo* algorithm suitable for generating dense digital terrain maps.

6. The **Projective Structure Recovery** module is the subject of future research into nontraditional algorithms for structure recovery. In particular, algorithms for *planar surface recovery via projective invariants* and *3D structure recovery via invariants* are currently being investigated.

1.2 Schedule

Below is a three-year schedule showing the significant milestones we expect to achieve during the course of the RADIUS contract.

FY93

- Test model-to-image correspondence matching
- Test robust pose recovery
- Evaluate vanishing point analysis

- Demonstrate model extension (point features)
- Begin porting code to RCDE

FY94

- Demonstrate model refinement
- Demonstrate model extension (line features)
- Demonstrate model extension (volumetric models)

FY95

- Demonstrate acquisition of initial site models
- Test model reconstruction based on invariance
- Symbolic extraction of building surface structure
- Quantitive evaluation of all algorithms
- Finish porting all code to RCDE

1.3 Design Philosophy

The UMass design philosophy emphasizes model-directed processing under a rigorous 3D perspective camera model. Information from multiple images is fused for increased accuracy and reliability, using a flexible set of modules under image analyst control. These design choices are motivated in the paragraphs below.

The UMass system is designed around the goal of building, using and maintaining geometric site models. This choice is based on the idea that **model-directed processing** provides a way to perform basic image understanding tasks more efficiently and reliably than purely bottom-up procedures. An example is model-to-image registration, where 3D features in a site model are automatically matched with 2D image features and used to compute camera pose, thus avoiding laborious hand-selection of accurate control points in the images. The very names of tasks like model extension and model refinement were chosen to emphasize the model-based nature of the UMass system.

Unlike several previous aerial image analysis systems that assumed nadir views only, the UMass system can handle oblique views as well. This is possible because the system is designed to perform a full **3D perspective analysis** of the scene. This allows the system to handle general viewpoints; indeed, many of the algorithms being used in this project are ported directly from the Unmanned Ground_Vehicle project, where they were used in the performance of terrestrial robot navigation. Oblique images, rather than being troublesome data to be avoided, are instead a boon to aerial photointerpretation. Sets of oblique images taken from widely disparate viewpoints tend to have large camera baselines and vergence angles, which lead to more accurate triangulation results.

Many systems deal with single, monocular images or a stereo pair of images, but the UMass system is designed to support **multi-image processing**. This design decision is based on the observation that incorporating information from many views at each stage of processing yields more reliable results. One example is a multi-image triangulation routine that combines corresponding 2D points across n images into a single 3D point estimate with a variance that is roughly proportional to $1/n^2$. Multi-image processing lends itself well to batch-style processing of several images; however it is also possible to formulate recursive versions that support incremental processing over time.

The **imagery analyst** plays a key role in the system being developed. Perhaps the most important task for the IA is determining what site features are important enough to be worth adding to the model. To this end, the IA can guide the model extension process by pointing to or outlining areas of interest that warrant detailed photogrammetric processing. The IA will also be of invaluable help for tuning a small set of operational parameters for each new domain of interest, and for interactively guiding the system through the initial stages of acquiring a new site model. Future IU research will focus on ways to further automate the site model acquisition process.

1.4 Report Overview

Each subsequent section in this document looks at one of the six modules listed in Section 1.1 in more detail. Progress to date on the module is reported, and sample results are shown where available.

2 Feature Extraction

A site model will invariably be specified at a much higher level of geometric abstraction than the pixel intensity values in an image. To help bridge the huge representational gap between pixels and site models, feature extraction routines will be available to produce symbolic, geometric representations of potentially important image features. Many types of geometric features can be extracted from incoming site imagery, including straight line segments, line pencils (sets of line segments that would projectively intersect at a single point if infinitely extended), rectilinear line groupings, curves, corner points, regions of homogeneous intensity, and textured areas. Our current model matching and extension algorithms rely exclusively on straight line segments and corner features. Extracted line segments are later grouped into line pencils by the vanishing point analysis module (see Section 3).

2.1 Straight Line Extraction

Two straight line segment extraction algorithms developed previously at UMass have been evaluated on the RADIUS imagery. The Burns algorithm [11] begins by labeling pixels in the intensity image according to coarsely quantized gradient orientation. A connected-components algorithm then aggregates sets of adjacent pixels with similar gradient orientation into line-support regions, i.e. pixel regions with an intensity surface that supports the presence of a straight line segment. Symbolic line segments are computed by intersecting a plane corresponding to the average intensity of the line-support region with a least-squares plane representing the underlying intensity surface in that region. Sample results are shown in Figure 1.

The second line extraction algorithm to be evaluated was the Boldt algorithm [10]. Sample results from this algorithm are shown in Figure 2. At the heart of the Boldt algorithm is a hierarchical grouping system inspired by the Gestalt laws of perceptual organization. Zero-crossing points of the Laplacian of the intensity image provide an initial set of local intensity edges. Hierarchical grouping then proceeds iteratively; at each iteration, edge pairs are linked and replaced by a single longer edge if their end points are close and their orientation and contrast (difference in average intensity level across the line) are similar. Each iteration results in a set of increasingly longer line segments.

The Burns line extractor runs much faster than the Boldt extractor because it makes fewer local decisions at each stage of processing and maintains fewer intermediate data structures. Indeed, a stripped down version of the Burns algorithm has been used as a fast line finder for real-time robot navigation experiments [18]. On the other hand, the Boldt line extractor generally produces more accurate results (compare Figures 1 and 2). In particular, the Burns algorithm occasionally produces line segments which are considerably skewed in orientation. This problem is caused by oddly-shaped support regions resulting from slow intensity gradient changes in the image. A promising solution to this problem has been devised and is currently being implemented. For the moment, however, the less accurate but speedy Burns algorithm is ideal for providing a "quick look" at new data by generating a set of line segments suitable for early stages of image understanding, such as delineating man-

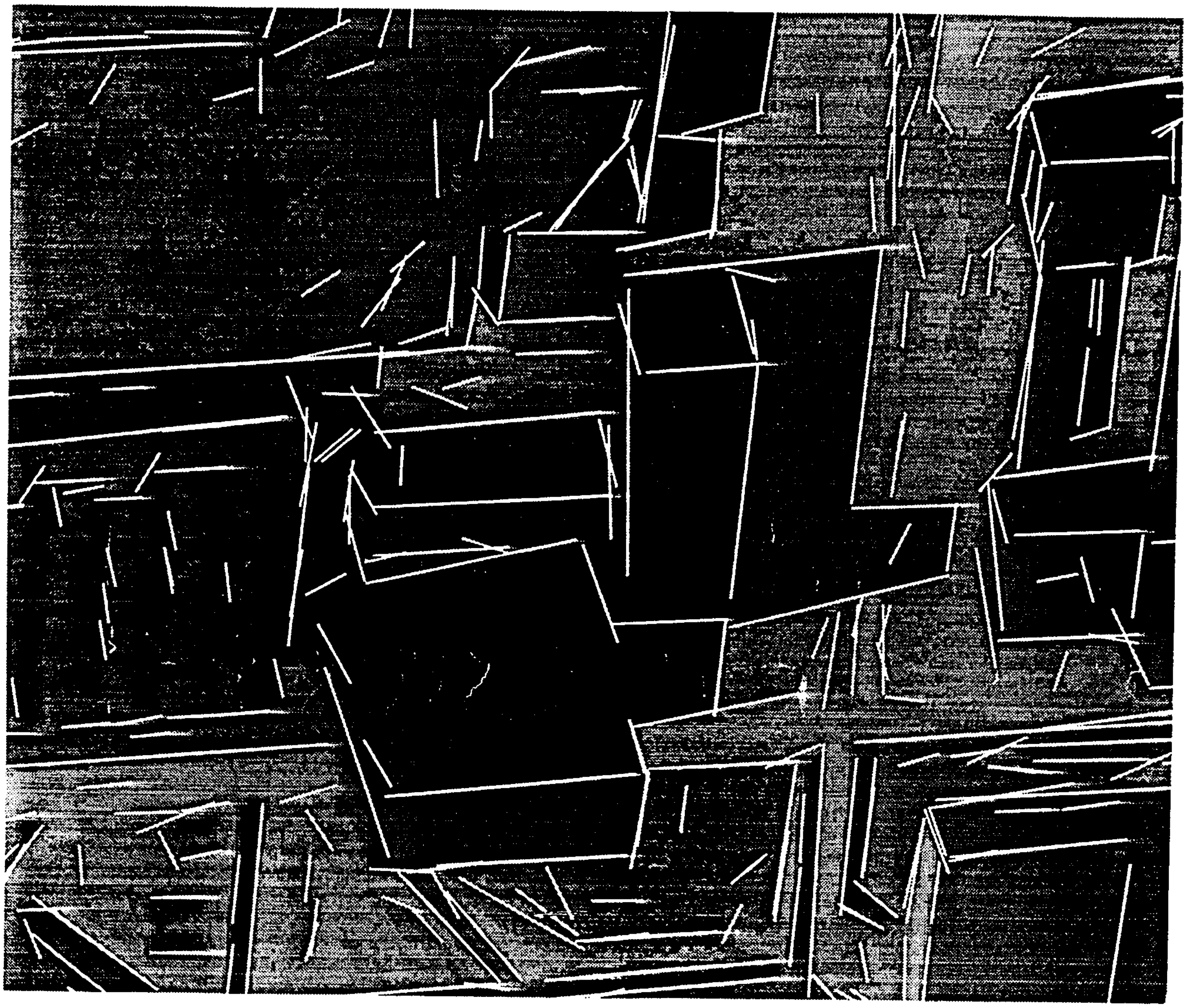


Figure 1: Straight line segments extracted by the Burns algorithm.

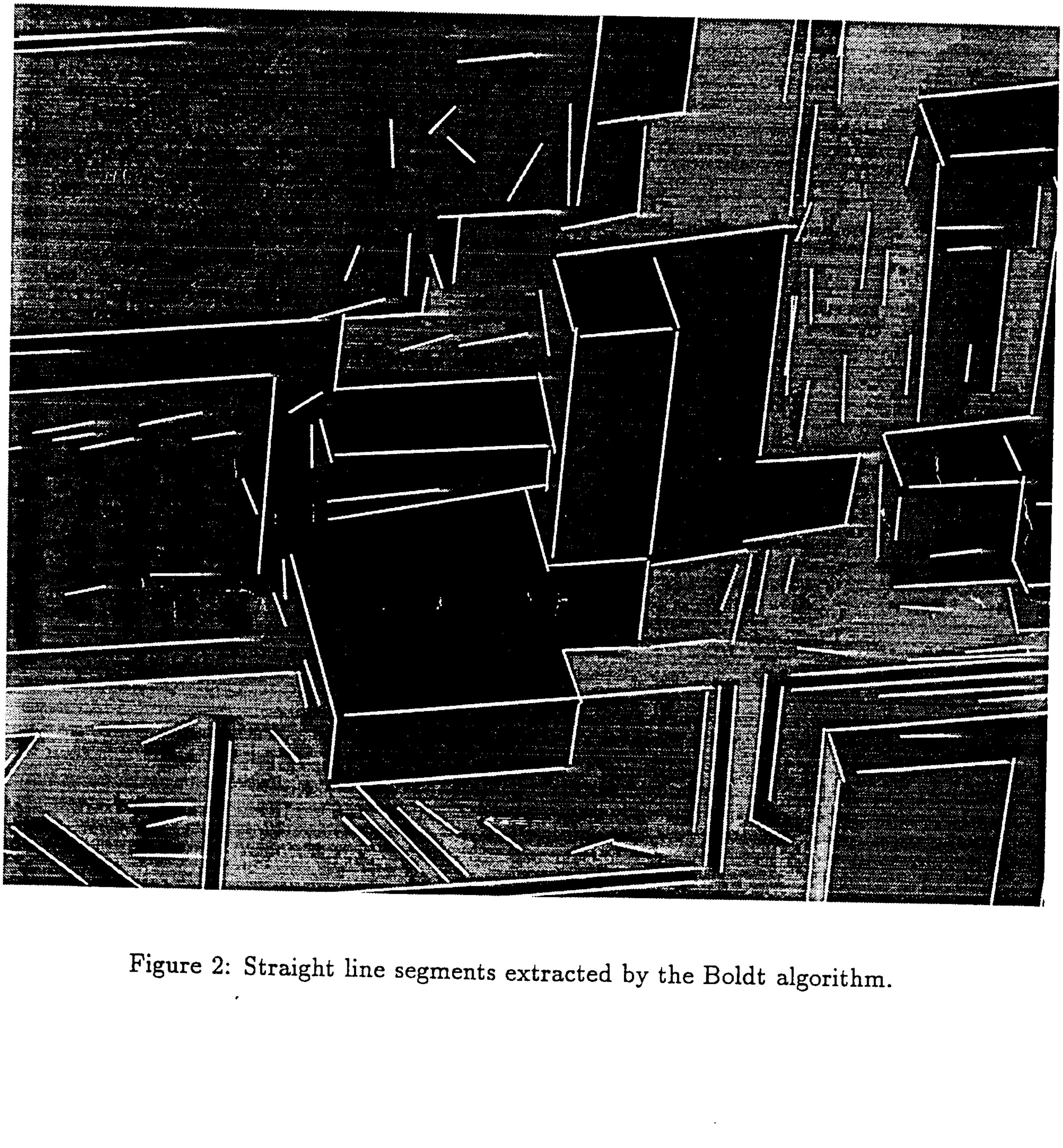


Figure 2: Straight line segments extracted by the Boldt algorithm.

made objects, computing initial estimates of camera orientation via vanishing point analysis (see Section 3), and even coarse model-to-image registration (Section 4). Meanwhile, the more accurate Boldt extractor is run off-line to provide the highly accurate line features needed for precise 3D geometric analysis and model construction.

Current implementations of both algorithms are unable to handle the 1300x1000 RADIUS modelboard 1 images in a single chunk. When processing such large images, several options are available:

1. use a focus of attention mechanism to determine subwindows in which lines will be extracted
2. break the image into subchunks that are processing and then pieced back together
3. reduce the size of the entire image via subsampling.

For our experiments using the modelboard imagery, a combination of the 2nd and 3rd options is used. First, image resolution is reduced by half using Gaussian filtering and subsampling. Each reduced image is then cut into a mosaic of overlapping subimages, each to be processed separately by the appropriate line extraction algorithm. All line segments found are translated and scaled back into their original image coordinate system, then filtered so that each line segment in the final set has a length of at least 10 pixels long and a contrast of at least 15 intensity levels. This procedure produces roughly 2800 line segments per image. As a side benefit of this strategy, Gaussian image reduction alleviates the peculiar "sawtooth" noise pattern that corrupts several of the modelboard 1 images.

2.2 Oriented Corner Extraction

A new oriented corner extractor was developed to automatically and accurately extract building corners. The corner extractor was designed to complement the straight line segment extractors previously described. Initial design requirements for the corner extractor were that it should:

1. be able to locate both dihedral and trihedral corners,
2. not be sensitive to building material and lighting direction (this rules out grey-scale template matching),
3. not be an excessive number of false positives in textured areas and
4. not be computationally expensive to run, since it will be used over large images.

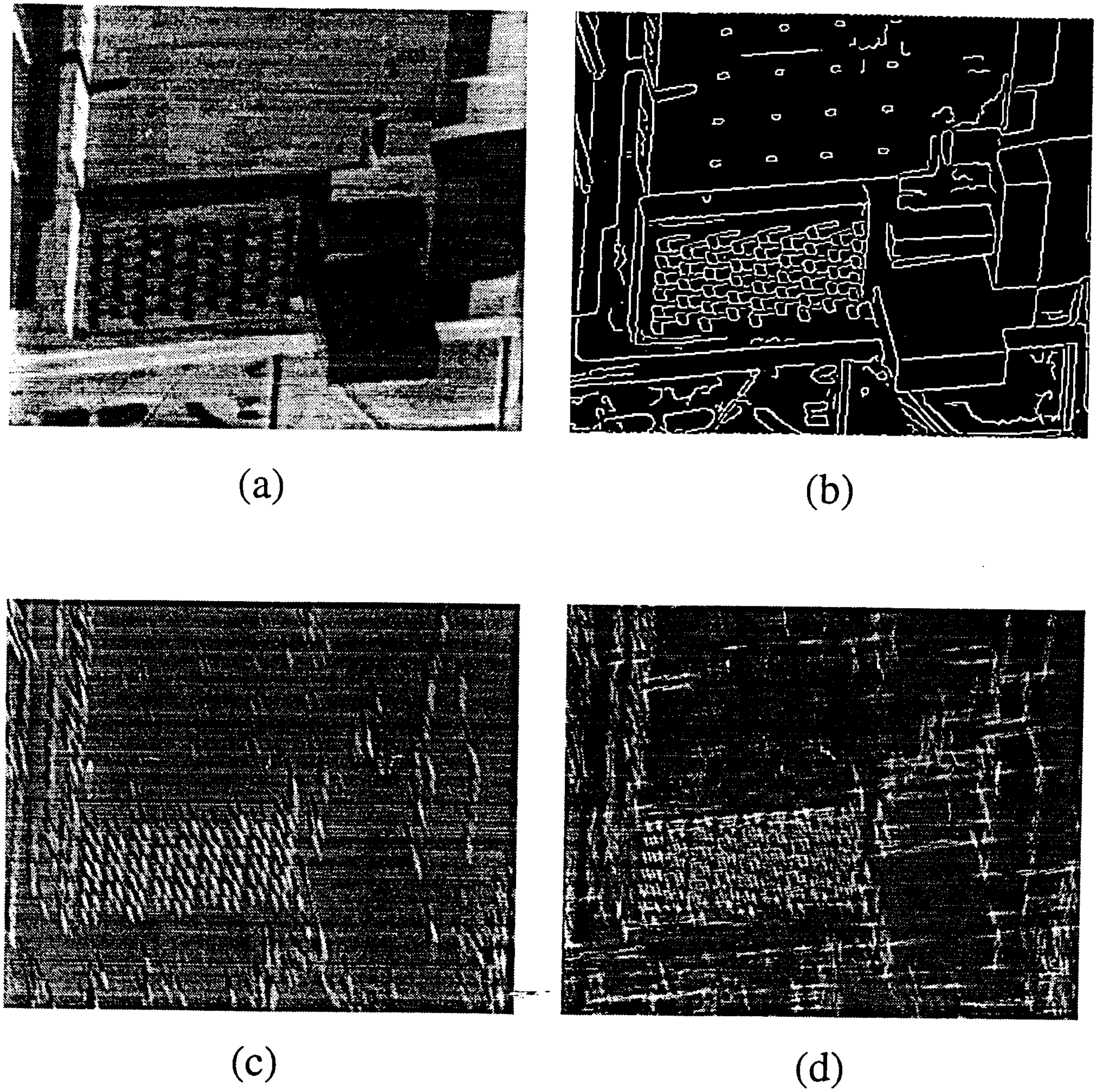
One common approach to finding corners is to search for two or more line segments with endpoints in proximity, then estimating the corner vertex as the intersection of the extended lines. This approach was rejected since we wanted an independent source of feature information that would complement the extracted line segments, not share their defects.

Instead, a two-stage method was developed: potential corners are detected using a set of oriented templates convolved with a Canny edge image, then the vertex position of each detected corner is estimated to subpixel precision using a least-squares fitting procedure. This two-stage approach yields a good balance of computation - the imprecise but fast convolution stage quickly generates a small (compared to the number of pixels in the image) number of candidate corner positions, then the more expensive, but highly accurate, least-squares estimation procedure is used to precisely locate each candidate corner.

For greater computational expedience, it is assumed that the expected orientation of lines in the image meeting to form a corner is known. This assumption is valid for corners of buildings oriented with respect to three predominant, mutually orthogonal directions, as occurs in many urban and industrial sites. In these cases, the expected orientation in the image of the mutually orthogonal edges forming a dihedral or trihedral corner can be determined automatically by vanishing point analysis (see Section 3). Although building edges with the same 3D orientation can project to 2D edges with different orientations at different locations in the image, this variation is predictable and often varies so slowly that the same oriented template is valid over large portions of the image.

Intermediate steps in the oriented corner detection algorithm are illustrated in Figure 3. First, given a greyscale intensity image (Figure 3a), a Canny edge image is formed (Figure 3b). Then, for each of the three dominant orientations in the scene, an oriented line endpoint template is built and convolved with the Canny edge image (the result of convolution with one oriented template is shown in Figure 3c.) Each template is designed to give a strong positive response at one end of a chain of edges having the correct orientation, a strong negative response at the other end, and a zero response everywhere else. The absolute value of each of the convolution images is then taken and the three are added together. The result is a "cornerness" image with strong peaks (Figure 3d) in the vicinity of dihedral and trihedral corners. For each peak attaining a given threshold, a corner hypothesis is formed.

Choosing a locally maximal peak in the "cornerness" image only localizes the vertex of a corner to within a pixel at best, and can be off by two or three pixels due to imprecision in the orientation of the convolution templates. To localize the vertex position to subpixel precision, a constrained least-squares procedure is performed. For each hypothesized corner, the Canny edges contributing to the hypothesis are analyzed to determine whether the corner is dihedral or trihedral, and in which directions the contributing edgel chains radiate. Two or three oriented lines constrained to meet at a single vertex are then fit simultaneously to the contributing Canny edges in a least-squares sense, and the position of the best-fit vertex is noted. Although the least-squares fitting procedure is moderately expensive compared to the initial corner detection phase, it is only run in places where there is already strong evidence that a corner exists. The final result of the corner feature extractor is a set of symbolic corner features consisting of a subpixel vertex estimate, and the number and orientation of image edges meeting at that vertex (Figure 4).



(a)

(b)

(c)

(d)

Figure 3: Illustration of oriented corner detection. (a) Gray-scale intensity image. (b) Canny edge image. (c) Results of convolving the Canny edge image with one oriented template determined from vanishing point information. (d) The final *cornerness* image. Local intensity peaks in this image correspond to dihedral and trihedral corner hypotheses.

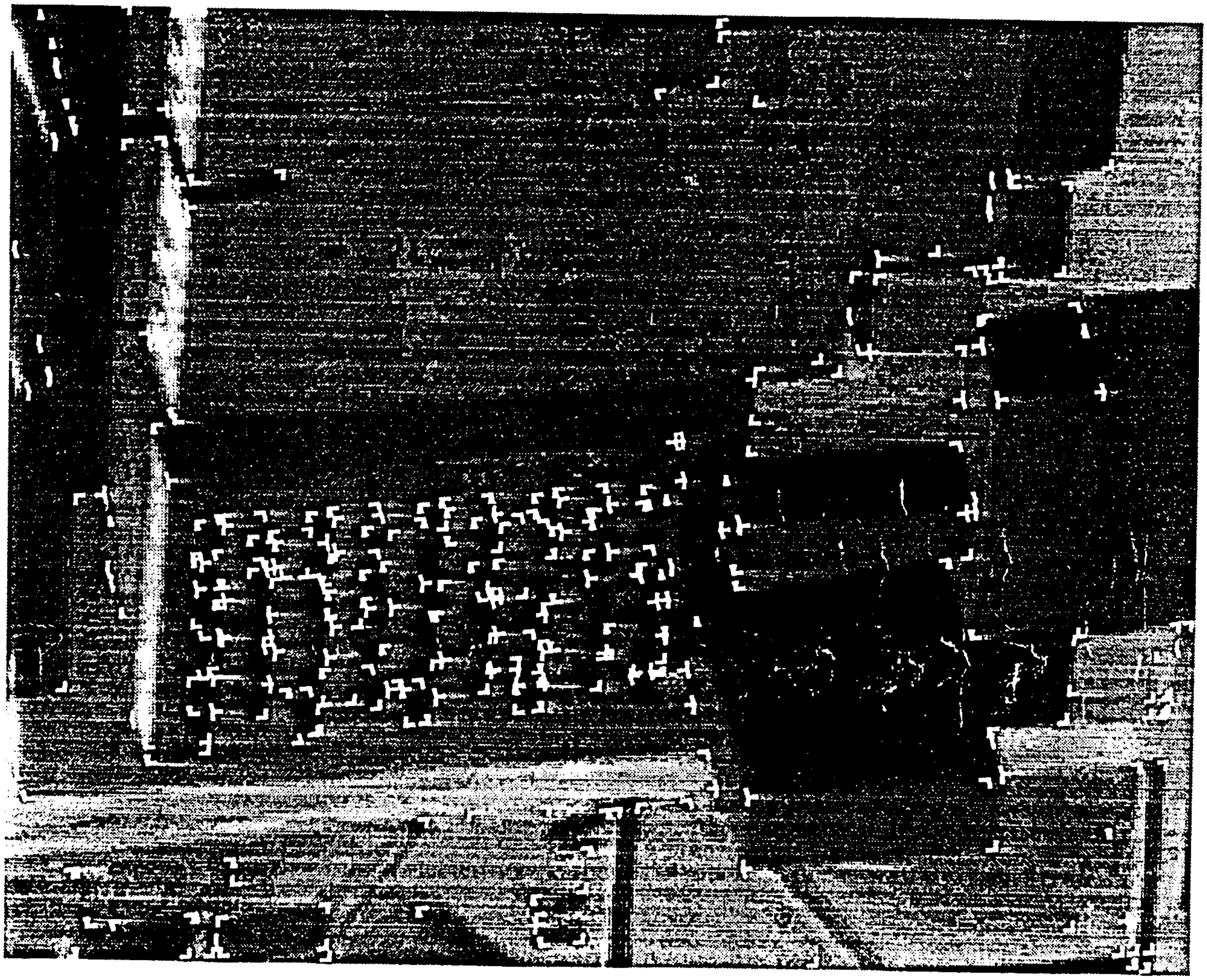


Figure 4: Final set of symbolic corner hypotheses. Corner vertices are located to subpixel precision using a constrained least-squares vertex fitting procedure.

3 Vanishing Point Analysis

Buildings in urban or industrial areas are often oriented with respect to an underlying rectangular grid (city blocks, for example). When viewed from the air, their roof lines appear to converge to two distinct vanishing points located on the horizon. For a nadir view, parallel 3D roof lines remain parallel in the image, and their vanishing points are then said to occur at infinity. A set of lines that projectively meet at a common point of intersection is called a *line pencil*. Under known camera lens parameters, vanishing point line pencils allow the computation of three-dimensional line and plane orientations from a single image, and thus allow the orientation of the camera to be determined with respect to the underlying local site coordinate grid. When the camera lens parameters are not known, they can be determined to a limited extent from vanishing point information [28].

A practical algorithm for estimating line orientations from vanishing points must address two issues: how to cluster line segments into pencils of lines passing through a single vanishing point, and how to accurately estimate 3D line orientations from these line pencils. Sections 3.1 and 3.2 discuss these topics in turn. An experimental evaluation of the accuracy of 3D line orientations computed from vanishing point information is reported in Section 3.3.

3.1 Vanishing Point Detection

Vanishing point detection involves finding clusters of lines directly towards a single point of intersection. A Hough-transform approach originally due to Barnard excels at quickly clustering line segments into intersecting groups [6]. Line segments in the image are mapped onto a histogram representing the surface of the unit sphere $x^2 + y^2 + z^2 = 1$ centered about the camera focal point. In practice, only the positive hemisphere $z > 0$ needs to be represented, and the surface of the hemisphere is partitioned by longitude and colatitude (see Figure 5). The sphere is a more appropriate histogram space than the image plane for detecting vanishing points because the sphere is a compact, finite surface, while the image plane is not.

Each line segment in the image, taken together with the camera focal point, forms a *projection plane* which intersects the unit (hemi)sphere in a great (semi)circle. Each histogram bucket maintains a count of the number of great circles passing through it; potential vanishing points are detected as peaks in the histogram, corresponding to areas where several great circles intersect. Barnard chose the center of the histogram bucket containing a peak as a point estimate of the vanishing point location.

We have modified Barnard's basic algorithm to support statistical estimation techniques that more accurately determine the true vanishing point location. The most fundamental change is that the histogram data structure is applied only as an initial clustering method and as an efficient spatial access mechanism, but the final analysis of vanishing point location is performed on the underlying line segment data. To achieve this, each histogram bucket maintains a list of the line segments that pass through it in addition to the count. After a peak is detected, the line segments within it are retrieved and statistical estimation techniques are applied to derive a more accurate estimate of vanishing point position (see

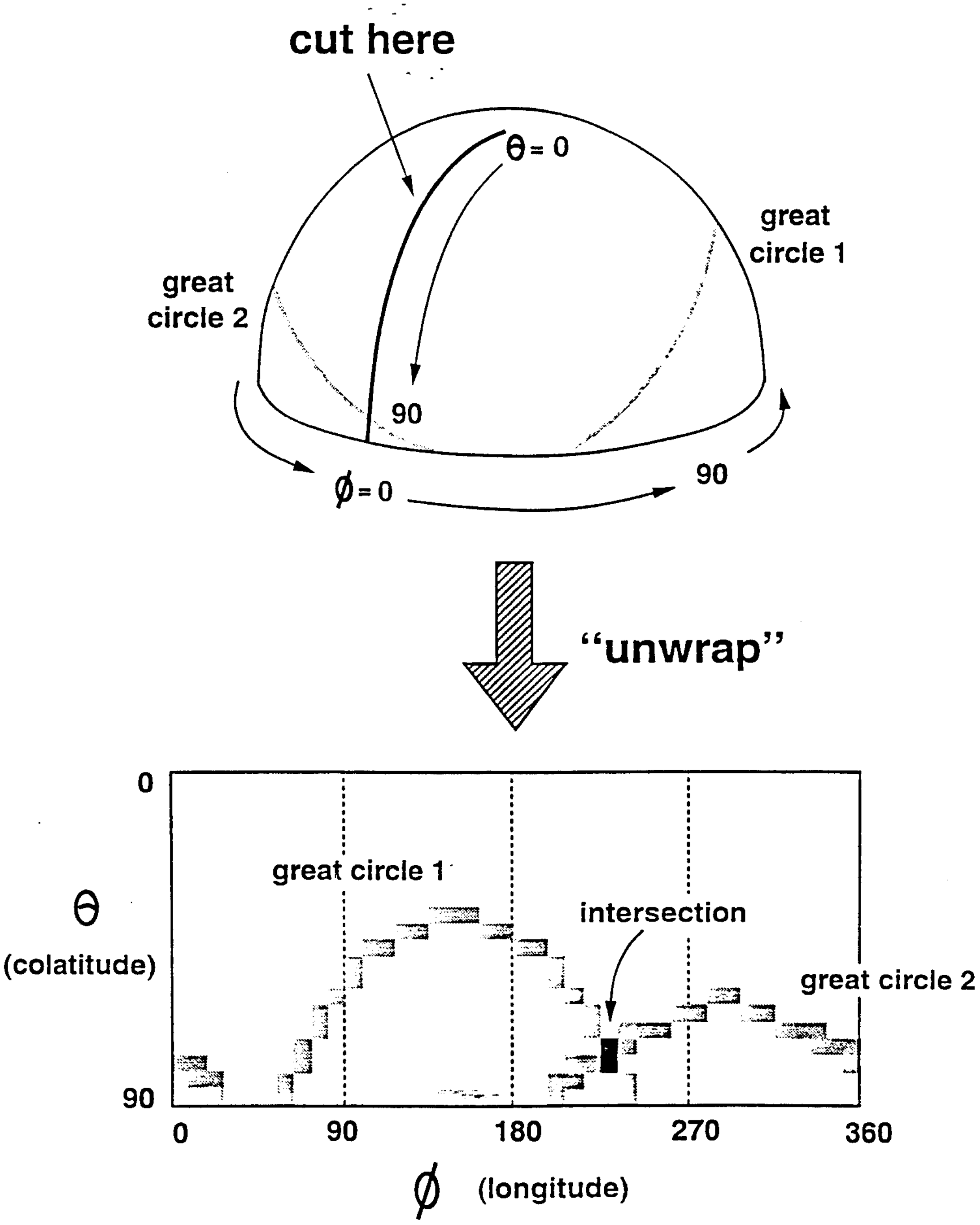


Figure 5: Barnard's histogram method for finding vanishing points. A hemispherical histogram is partitioned by longitude and colatitude. For each line segment in the image, a great circle of histogram cells is incremented. Potential vanishing points are detected as peaks in the histogram, corresponding to areas where several great circles intersect.

Section 3.2).

Peak selection in Hough-transform methods is inherently problematic [21]. In particular, when the true position of a vanishing point on the sphere falls near a histogram boundary, candidate line segments that should be grouped together fall into separate buckets. For this reason we actually collect line segments from a whole neighborhood of buckets surrounding the peak. We have implemented a routine to visit every grid cell falling within a circle of given radius ρ from a given point on the hemisphere, taking into account the wraparound that must occur due to the mapping between unit hemisphere and 2D rectangular array of grid cells.

After the largest cluster of converging lines is detected from the histogram, all lines contained within that cluster are deleted from the histogram, and all bucket counts are updated. The highest remaining peak is taken as the second cluster, and so on. Multiple vanishing point peaks are thus detected in decreasing order of number of lines contributing to them. The number of vanishing point clusters to look for is a user-supplied parameter – usually less than three appear in any image.

An illustration of vanishing point detection on model board imagery is shown in Figure 6. Two line pencils are detected, with roughly 900 line segments in each pencil. These correspond to the two dominant sets of horizontal line segments in the scene. Detecting two sets of scene horizontals via vanishing point analysis, and thus finding the horizon line of the ground plane in the image, has proven relatively easy in the model board images. Detecting the third, vertical vanishing point is not so easy, because the associated image lines are fairly short and tend to fall below the line length filter we apply in an effort to keep the amount of line data more manageable. When needed, we compute the vertical vanishing point analytically from the two horizontal vanishing point directions. The vertical vanishing point computed in this way can be used to search for short lines in the image that correspond to vertical edges in the scene.

3.2 Statistical Estimation of Vanishing Points

While Barnard's histogram method excels at quickly clustering line segments into convergent groups, a final estimate of vanishing point location and variance should be based on the line segments themselves rather than the arbitrary bucket boundaries of a histogram data structure. We therefore use the histogram mechanism only for clustering and efficient spatial access, while estimating vanishing point location using a geometric statistical procedure. Because the vanishing point may be at infinity in the image plane, the quantity actually estimated is a unit vector pointing towards the vanishing point. The orientation of this unit vector is also an estimate of the 3D orientation of parallel lines having that particular vanishing point.

Collins and Weiss [12] present a formal statistical approach to estimating line orientations from vanishing points. Assuming that image line segments have been previously grouped into line pencils, the projection plane normals for each pencil form a set of points on the sphere clustered about a great circle perpendicular to the true vanishing point orientation vector (see Figure 7). Collins and Weiss treat this cluster as a random sample from an equatorial

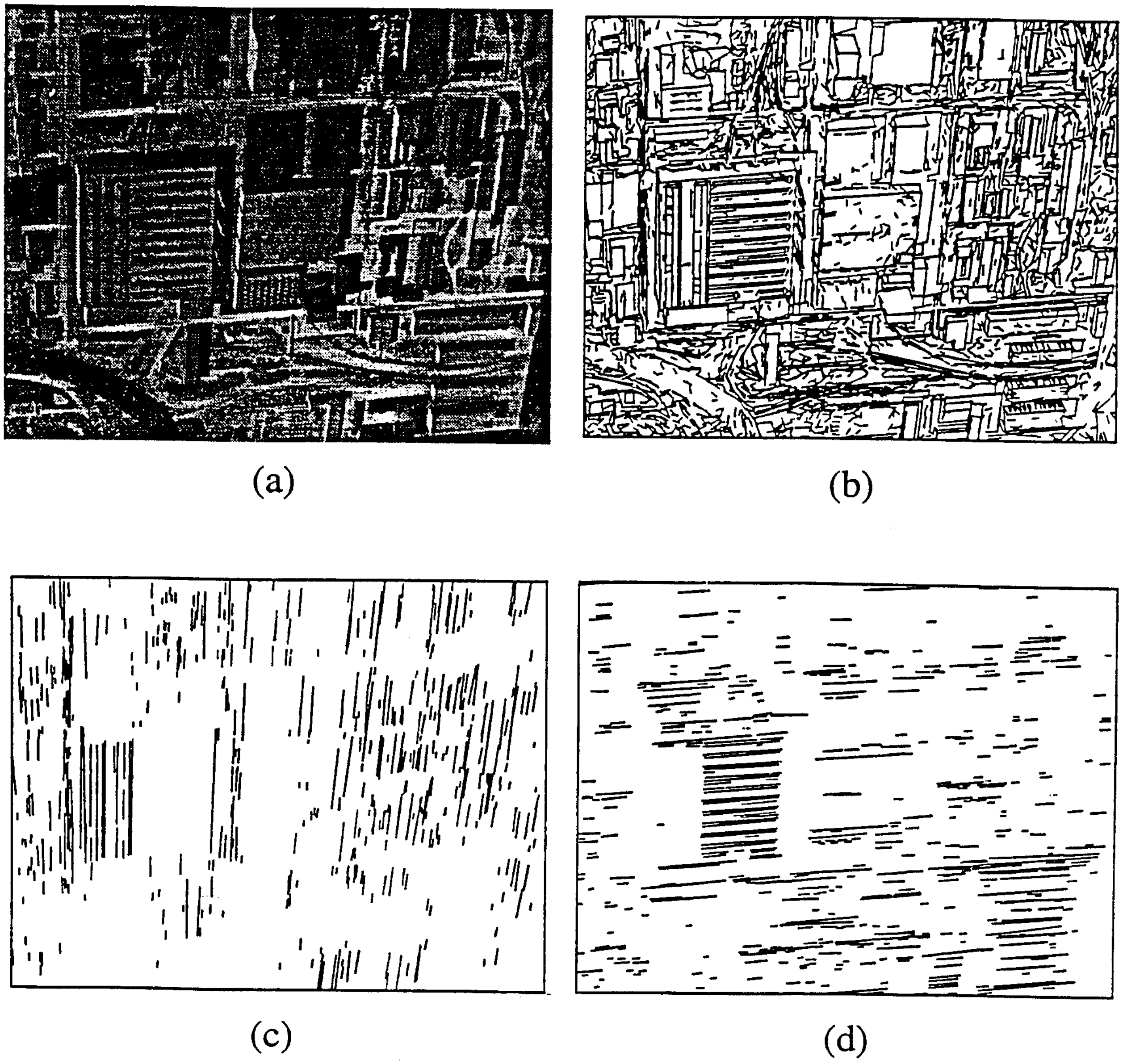


Figure 6: An example of vanishing point detection. (a) Modelboard 1 image J8. (b) Straight line segments extracted by the Burns algorithm (see Section 2.1). (c) and (d) The two largest line pencils found using the vanishing point clustering algorithm described in the text. These line pencils correspond to sets of parallel 3D line segments in the scene.

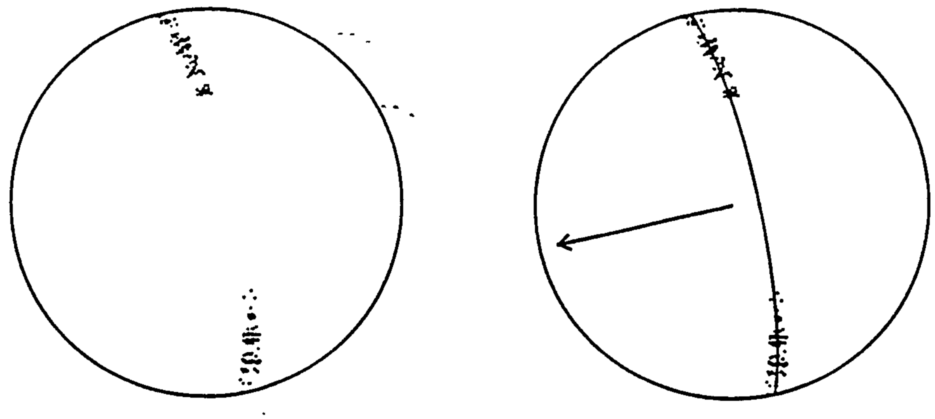


Figure 7: Vanishing point estimation on the sphere. The projection plane normals of line segments in a vanishing point pencil cluster around a great circle on the unit sphere. The polar axis of the best-fitting great circle provides an estimate of vanishing point location in the image, and of the 3D orientation of parallel lines associated with that vanishing point.

probability density function on the sphere, and estimate vanishing point orientation as the *polar axis* of the density function using standard maximum-likelihood estimation techniques.

Although it works quite well in practice, the maximum likelihood estimator of Collins and Weiss has one main shortcoming. All lines in a line pencil are treated equally, contrary to practical experience which shows that longer lines are detected more accurately than shorter ones, and thus should be given more credence when estimating line intersections. A new statistical estimator has now been developed, based on Bayesian estimation of the axis of a great circle of noisy observed points. The unknown true values of each point are assumed to be constrained to lie exactly on a great circle, and each observation is assumed to be perturbed by an independent probability density function on the sphere with a local covariance computed from a simple random perturbation error model on the associated image line segment. This has the effect of assigning smaller variances to longer lines in the image, and hence assigning them a higher weight in the resulting estimation process.

3.3 Evaluating the Accuracy of Orientation Estimates

An experimental evaluation of the effectiveness of the new vanishing point orientation estimator was carried out on images J1-J8 from the modelboard 1 data set. The goal of this experiment was to compare 3D line orientations computed by the statistical estimator against absolute ground truth orientations. The ground truth orientations were provided by Lynn Quam at SRI International, who computed them using a multi-image, block adjustment procedure [3] where the camera pose parameters, effective camera focal length, and principle point (image center) for all eight images were computed simultaneously. It should be noted that the line orientations derived from Quam's computed camera pose values are themselves, only estimates of the true relative line orientations. However, since Quam's computations are based on precisely measured 3D ground and 2D image control points, they

Table 1: Absolute orientation errors in estimated vanishing point orientations, and in the relative angle between estimated orientations.

Image	Abs Err in VP1 (deg)	Abs Err in VP2 (deg)	Rel Ang (deg)
J1	3.7	1.0	91.2
J2	3.7	2.8	91.7
J3	1.5	1.1	89.5
J4	4.7	0.5	87.3
J5	1.9	0.5	91.1
J6	2.4	2.3	90.3
J7	1.9	6.8	87.7
J8	2.0	3.5	91.9
avg	2.7	2.3	90.1
std	1.1	2.1	1.8

are expected to be inherently more accurate than vanishing point estimation methods that are based only on generic knowledge about parallelism in the scene.

The experiment was carried out as follows. For each of eight images J1-J8, a set of line segments was produced using the Burns straight line extraction algorithm (Section 2.1). This procedure produced roughly 2800 line segments per image. Barnard's method was then used to histogram the set of line segments from each image, and the two largest line pencils were extracted (e.g. Figure 6). In all cases, the two largest line pencils in each image corresponded to the two dominant orthogonal sets of lines in the 3D scene. Following extraction of line pencils, the vanishing point orientation estimator was applied, and the resulting 3D line orientation estimates were compared against the ground truth relative orientations for north-south and east-west line segments in the scene.

Table 1 tallies the results of this comparison. For each image, absolute errors between estimated and ground truth line orientations for the two dominant orthogonal vanishing points in the scene are compared, along with the computed relative angle between the estimated orientations. Sample means for the absolute errors in the two vanishing point orientations are 2.7 and 2.3 degrees. The average relative angle between the two estimated vanishing points orientations, which in reality is 90 degrees, was computed as 90.1 degrees with a standard deviation of roughly 2 degrees.

We note in passing that although the model board scenes are ideal for vanishing point detection, in the sense that there are many parallel lines in two distinct orientations, they are among the harder cases for accurate vanishing point estimation since the convergence angle of line segments across the image is never very large. In these images each vanishing point is significantly off the image plane, and in many cases, is far off, near infinity. The benefits of computing line intersections on the sphere rather than the extended image plane are very apparent in these cases.

4 Model-to-Image Registration

Matching and registration is a ubiquitous problem in computer vision. Correspondence matching can be broken into two general areas: *model-to-image* registration where correspondences are identified between known 3D model features and their 2D counterparts in an image, from which the absolute pose of the camera is computed, and *image-to-image* registration where corresponding features in two images of the same scene must be determined, and from them the relative pose of the image pair. Image-to-image registration will be covered in Section 6.

Accurate model-to-image registration is a necessary precursor for many site modeling tasks. Proper registration between an incoming image and a stored geometric site model determines the position and appearance of important model features in the image. The model can then be overlaid on the image to aid visual change detection and verification of expected scene features. Model-to-image registration using templates for movable objects such as trucks and railway cars provides a method for locating and counting these vehicles. Finally, correspondences determined by matching 3D model features with 2D image features, using an initial estimate of camera pose, can be used in conjunction with a *pose refinement* algorithm to recover a more accurate estimate of camera pose.

The model-to-image registration process involves two tasks: 1) *correspondence matching* to determine correspondences between model features and image features, and 2) *camera resection* to determine the precise geometric relationship between the image and the scene. The first task is by far the hardest, and its success depends on finding a good initial set of model-to-image correspondences automatically. Whether a good set of initial correspondences can be found easily or not depends on the quality and completeness of initial estimates of the image acquisition parameters. As a general rule-of-thumb, the difficulty of finding correspondences is directly proportional to the number of unknown acquisition parameters (unconstrained degrees of freedom in the model-to-image transformation space) and the amount of uncertainty in the acquisition parameters that *are* known. Correspondence matching is discussed in Section 4.1.

The second aspect of model-to-image registration is camera resection. It is important to note that since model-to-image correspondences are being found automatically here, subsequent camera resection routines need to take into account the possibility of mistakes or *outliers* in the set of correspondences found. This implies that the camera resection routines need to use robust estimation procedures that are impervious to the effects of outliers. When the internal orientation or "lens" parameters of the camera are accurately known, and only the the external orientation or "pose" parameters need to be precisely determined, the camera resection process reduces to a pose estimation problem. A robust pose estimation procedure has been developed and tested at UMass, and it will be described in Section 4.2. We conclude in Section 4.3 with an experimental evaluation of model matching and pose determination using a pair of images from the Martin Marietta, Denver, Colorado site.

4.1 Model Matching

The goal of model matching is to find the correspondence between 3D features in a site model and 2D features that have been extracted from an image. For example, we currently use site models that are composed of wireframe buildings; model matching in this case involves determining correspondences between edges in a 3D building wireframe and 2D extracted line segments from the image. To find this correspondence, we are evaluating a model matching algorithm developed at UMass by Ross Beveridge [7]. Based on a *local search* approach to combinatorial optimization, this algorithm solves for the correspondence between model edges and image line segments, and for the transformation that brings the projected model into the best possible geometric alignment with the underlying image data.

The local search matching algorithm searches the discrete space of correspondence mappings between model and image features for one that minimizes a match error function. The match error depends upon the relative placement implied by the correspondence, and the amount of coverage of the model by the data. That is,

$$E_{\text{match}} = E_{\text{fit}} + E_{\text{omission}} \cdot \tag{1}$$

In particular, fit error is computed by choosing pose parameters that cause projected model edges to appear most similar to the image edges currently hypothesized to be in correspondence with them. The similarity between image edges and projected model edges is measured by a least-squares residual fit error. The omission portion of the error term is computed as the percentage of projected model lines having no corresponding data lines to explain them. The mathematical transformation that maps model features into the image is essentially a module of the system. Our current implementation handles the four parameter 2D similarity transform and the full 3D pose transform. These options are described more fully in Section 4.1.2.

To find the optimal match, probabilistic local search relies upon a combination of iterative improvement and random sampling. Iterative improvement refers to a repeated generate-and-test procedure by which the algorithm moves from an initial match to one that is locally optimal via a sequence of incremental changes (addition or removal of a model-data correspondence pair) that continually reduce the match error. In an effort to find the global optimum, the algorithm is run multiple times starting from different (random) initial correspondences in the model-to-data line match space. Even if the probability of seeing the optimal match on a single trial is low, the probability of seeing the optimal match in a large number of trials, started from uniformly random positions in the match space, is high.

Figure 8 presents a pictorial vignette of the model matching process. Given an incoming image (8a), the first step is to extract a set of 2D symbolic image features (8b) – in this case, straight line segments extracted by the Burns algorithm [11]. Also provided is a 3D wire-frame model (8c) containing prominant site features whose locations are known with respect to the local site coordinate system. This model will be registered with the image, and in so doing, the position and orientation of the camera in the local site coordinate system will be determined. The 3D model is projected into the image usir an initial, rough estimate of the camera pose parameters (8d), and a set of candidate correspondences for each projected wireframe edge is selected from the underlying image line segments (8e). The model

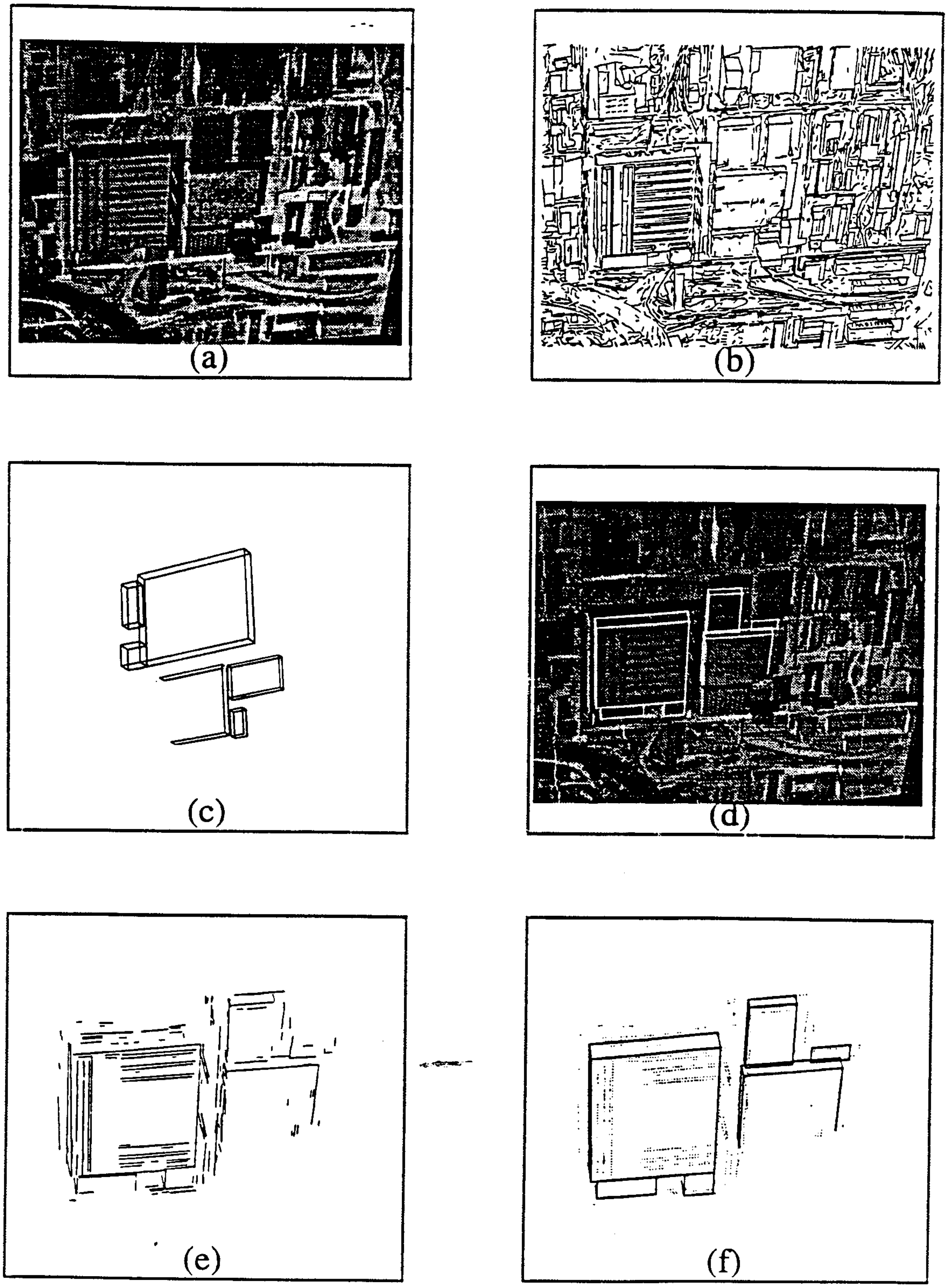


Figure 8: A pictorial summary of the model matching process. (a) Model board image J8. (b) Burns lines for image J8. (c) Partial wire-frame model. (d) Initial model projection. (e) Candidate data correspondences. (f) Best match of model to data.

matcher then searches through this space of possible model-to-image feature correspondences to determine the best set of matching features. The optimal correspondence found is used to estimate the camera pose that best brings projected model features into alignment with their corresponding image features (8f).

The following subsections explore some of the practical issues involved in applying the local search model matching system. Topics treated include initial estimation of the image acquisition parameters, selection of a set of possible initial correspondences, the trade-off between efficiency and accuracy when choosing the number of degrees of freedom in the model-to-image transformation space, and use of a hierarchical matching strategy to further reduce the combinatorics of correspondence matching.

4.1.1 Setting up the Match Space

Initial Estimates of Acquisition Parameters

The unconstrained correspondence matching problem is still an unsolved research problem in computer vision. As a practical matter, initial estimates of the image acquisition parameters need to be available beforehand to cut the enormous number of potential matches down to a manageable level.

We model the image acquisition process as a projective transformation from the 3D scene onto a 2D image plane. Using homogeneous coordinates, this transformation can be represented by a 3×4 matrix of parameters, defined up to a single scale factor. Thus, although there are 12 entries in the transformation matrix, there are only 11 independent parameters. These parameters can be further subdivided into 5 internal (lens) parameters, and 6 external (pose) parameters, and may be arranged in the following matrix formula,

$$\begin{array}{cccccc} & 3\times 1 & & 3\times 3 & 3\times 4 & 4\times 1 \\ k & \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} & = & \begin{bmatrix} s_u & \alpha & u_0 \\ 0 & s_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} & \left[\mathbf{R} \,\middle|\, \mathbf{t} \right] & \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \end{array} \qquad (2)$$

which maps the 3D coordinates x, y, and z of a model point into the 2D coordinates u and v of an image point. Here, the transformation from model to image coordinates is broken into two stages, a 3×4 matrix representing camera pose and the process of perspective projection, followed by application of a 3×3 matrix of camera lens parameters. The pose parameter matrix is further partitioned into a 3×3 orthonormal rotation matrix $\boldsymbol{R}$ (containing 3 degrees of freedom) and a 3×1 translation vector $\boldsymbol{t}$ (containing the remaining 3 degrees of pose freedom). The 5 camera lens parameters are represented by s_u and s_v, which are the focal lengths in pixels along each of the image axes, u_0 and v_0, which are the pixel coordinates of the camera principle point, and an image axis skew parameter α, which is always assumed to be 0 unless information to the contrary exists.

For our experiments, initial estimates of the camera lens and pose parameters for the RADIUS model board imagery were determined as follows. First, nominal values for the internal camera parameters were filled-in from information supplied with the model board data (namely 6.8 micron square pixels and a focal length of 47.9 mm for the 18 inch GSD images) and by assuming the principle point to be in the numeric center of the image. To estimate the external camera parameters, the orientation of the camera with respect to the scene was first determined by vanishing point analysis (see Section 3), up to a four-fold ambiguity that was resolved by identifying the direction of true north in the image by hand. The distance of the camera from the ground was computed from the reported Ground Scale Distance (GSD); to date our experiments have only used the 18 inch GSD images. Finally, the intersection of the camera's line of sight with the ground plane was estimated manually.

Selecting the Correspondence Space

Model matching performs a search through the space of possible model to data correspondences. This space is initially set up by deciding which data lines in the image are to be considered as candidate matches for each model line. Careful pruning of this space at the start is crucial to achieving tractable run times. The problem is greatly simplified when good initial estimates of the image acquisition parameters are available. The better the initial estimates, the tighter the filters for picking out possible candidate data lines can be, both in terms of orientation, position in the image, and length.

Even though the metric used to score potential correspondences is purely geometric (Equation 1), photometric expectations such as the sign and magnitude of contrast across a line can be enforced by prefiltering for these properties in the initial candidate generation phase. Our tendency has been to underspecify rather than overspecify filter parameters, however, because once a correct line pairing has been excluded by overzealous filtering in the candidate generation stage, that correct pairing can never contribute to the match that is eventually found.

For the experiments we have run on the model board imagery, we project each model line into the image using the initial estimate of image acquisition parameters described above. All image line segments having an orientation within 10 degrees of the projected model line orientation, and located within 100 pixels laterally from it, are then selected as possible matching candidates.

4.1.2 Model-to-Image Transforms

Essentially all model-to-image correspondence problems involve solving for a discrete correspondence between model and image features along with an associated transformation mapping model features into the image. The two problems together constitute model matching: a match being a correspondence plus a transformation. The most general transformation typically considered involves full 3D pose: a rigid 3D model is rotated and translated relative to the camera and then projected into the image using a known camera model. This amounts to solving for the pose parameters in Equation 2, while holding the lens parameters fixed.

Unfortunately, this approach to matching is very expensive computationally. To see why, consider the match error metric introduced in Equation 1, which is a combination of geometric fit and omission terms. To compute these terms for a particular set of model-to-data correspondence pairs involves solving for the transformation that brings projected model features into best alignment with their hypothesized corresponding data lines, where best is defined as minimizing the residual least-squares geometric fit error. Thus, to evaluate each state in the space of potential correspondence matches, a full 3D-to-2D pose solution must be computed. There is no known closed-form solution to the pose problem for an arbitrary number of points and lines – exact solutions require iterative, nonlinear least-squares techniques. In a typical run, the model matcher searches through thousands of possible match states in its search for the optimum correspondence. If at each match state one needs to solve the full 3D-to-2D pose problem, the resulting algorithm is prohibitively expensive.

There is great advantage, then, in looking for alternative methods to replace solving the full 3D-to-2D pose equations exactly at each step. We have investigated two such approaches, both originally introduced by Beveridge [8]. These methods are four-parameter 2D-to-2D affine (similarity) matching, and hybrid 3D-to-2D matching.

2D-to-2D Matching

The idea behind 2D-to-2D matching approximations to the full 3D-to-2D correspondence problem, is that given good initial estimates of the lens and pose parameters of the camera, the 3D model can be projected onto the image before matching begins, turning the problem into a search for the 2D transformation that best brings the *2D projected model features* into correspondence with the image data features. This is the underlying motivation for the 2D similarity version of the model matcher, which seeks the best four parameter affine or *similarity* transform relating a projected set of wireframe model edges with a set of image data line segments [7].

The 2D similarity transform is a planar transformation consisting of four parameters: a rotation angle in the plane, a 2 parameter translation in the plane, and a single global change of scale. It is a subgroup of the more general 6-parameter affine transformation group. Finding the best 2D similarity transform between a set of corresponding model and data lines is much faster than solving for full 3D pose. Indeed, Beveridge devised a closed-form solution for determining the similarity transform that brings a set of 2D model lines into alignment with their corresponding data lines. Because the similarity transform matcher is fast, it is possible to run more trials in a given amount of time, thereby increasing the confidence in finding the best correspondence.

Problems with 2D-to-2D Matching

We have run numerous matching experiments using the 2D similarity matcher on images J1–J8 of the Model Board 1 data set, using a simple wireframe model generated from the 3D ground truth data points provided. Initial estimates of the camera pose parameters were determined as previously described, then purturbed along their various degrees of freedom by hand. Our experiments showed that the performance of the 2D-to-2D similarity version of the matcher was sensitive to the accuracy of the initial camera pose. This was not entirely unexpected – a good initial pose estimate is crucial for 2D-to-2D matching since the initial pose determines what 2D projected model is matched against the data. Once it has been projected using the initial pose, the resulting 2D model can thereafter only be rotated, translated and scaled in 2D in an effort to fit it properly to the image. We had, however, expected this version of the matcher to work relatively well in the aerial domain, where the camera is far from the objects being viewed, as opposed to "close-range" domains, such as hallway navigation, where small differences in viewpoint can cause large changes in object appearance.

Further study revealed that the 2D similarity matcher was particularly sensitive to the accuracy of the initial estimate for the camera look vector. The reason for this can be explained by considering the relationship between similarity transformations of a 2D model in the image and the effects of the six degrees of freedom of camera pose on the initial projection of the model. Errors in the initial camera pose estimates lead to projected model templates that deviate from their ideal 2D location, orientation, size, and shape in the image. The four parameter similarity transformation can correct for errors in 2D image location caused by incorrect specification of the horizontal location (e.g. latitude and longitude) of the camera above the earth, errors in image rotation due to misspecification of the amount of rotation of the camera about the line of sight, and global image scale changes caused by incorrect specification of the distance of the camera from the scene. The 2D similarity transform cannot recover from errors in image object shape, however. These shape errors are due to incorrect estimates of the direction and amount of object foreshortening, and are a direct result of errors in the initial orientation estimate of the camera look vector.

Figure 9 illustrates an example of the problem that can occur due to incorrect estimates of foreshortening. Figure 9a shows an initial projected model template overlayed on one of the model board images. The most notable discrepancy at this stage is a large error in translation of the 2D model template with respect to its correct position in the image. Figure 9b shows the results of applying the best four parameter similarity mapping found by the 2D local search matcher, and Figures 9c and 9d are zoomed images showing the match in greater detail. The overlap between the 2D model template and the image has been greatly improved – a four parameter similarity transformation has been found that accurately aligns the model template on three out of four sides of the main building. However the fourth side, shown in detail in Figure 9d, is still considerably misaligned. It has to be stressed that this is not a failure of the 2D matching algorithm, *per se*. It has found the best possible match allowed under these circumstances. The problem is that no combination of 2D rotation, translation and uniform scaling can possibly bring the initial projected 2D model into alignment with the underlying image, because of the excessive foreshortening in one direction of the model template due to an inaccurate initial estimate of the camera look vector.

Hybrid 3D-to-2D Matching

We have observed the foreshortening problem described above in many of our 2D-to-2D matching experiments on the model board images. This has led us to abandon the 2D similarity matching system in favor of a hybrid 3D-to-2D perspective matching system also developed by Beveridge [9]. This system combines the speed of 2D similarity matching with the accuracy of full 3D pose matching. In particular, the hybrid matcher interleaves 2D similarity matching with 3D pose updates. Starting with an initial projection of the 3D model to form a 2D model template, a full hill-climbing cycle of the 2D similarity matching

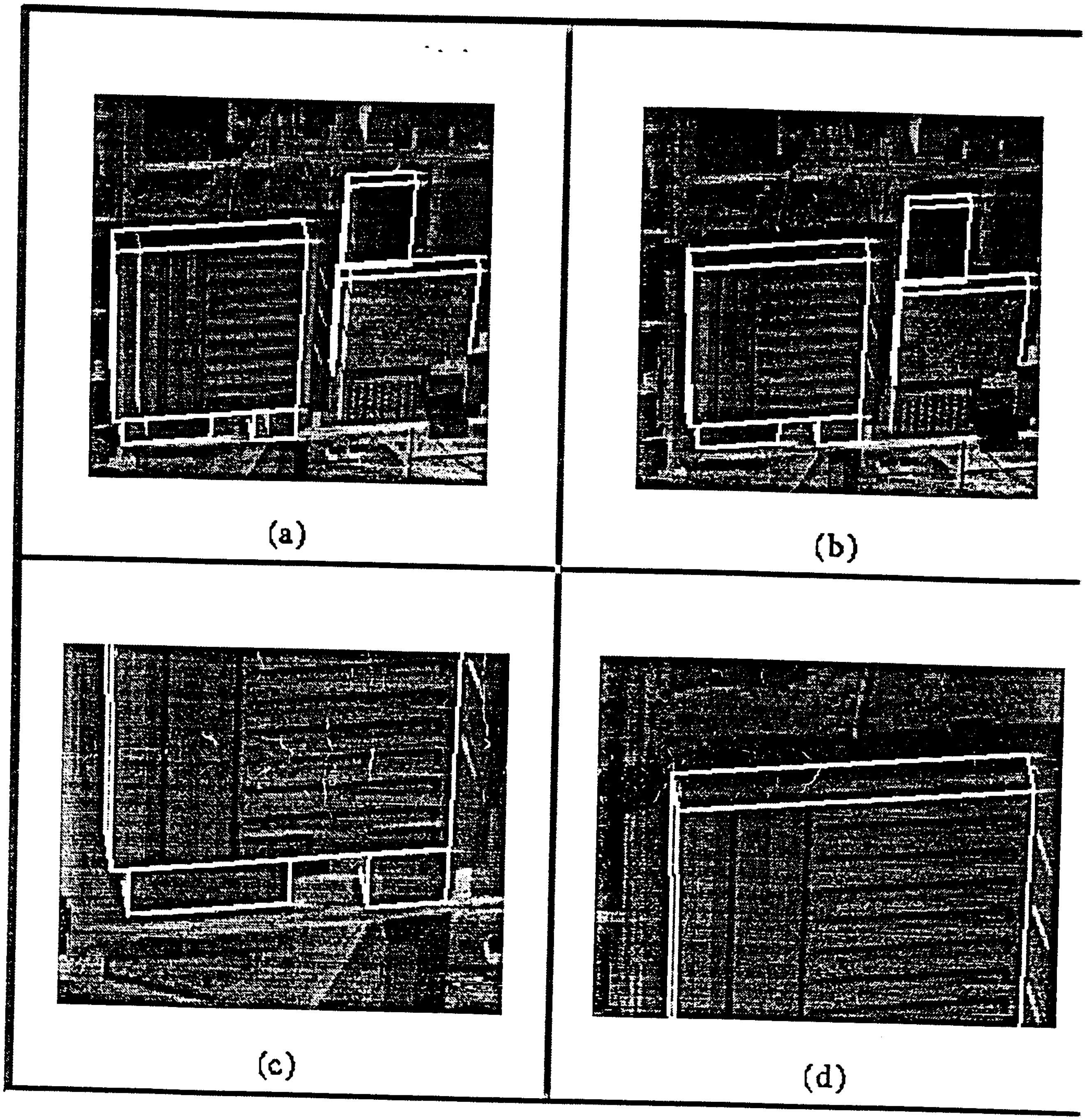


Figure 9: Model matching using the 2D similarity transform, illustrating the effects of foreshortening errors. (a) Initial projection of model template. (b) Overlay after 2D similarity model matching. (c) Close-up showing good alignment on three sides of the model. (d) Close-up showing poor alignment on the fourth side, due to uncompr~~n~~sated foreshortening errors.

system is performed from a random start in correspondence space. Once a local optimum is reached, the final correspondence between 3D model lines and 2D image line segments is passed to a full 3D pose determination algorithm (see Section 4.2). The computed pose provides a new estimate of camera location and orientation. This is used to reproject the 3D model to create a new 2D model template, and 2D local search similarity matching begins anew. The process converges when no change to the set of model-to-image correspondences can be found that improves the match score.

The crucial difference that allows the 3D-to-2D hybrid matcher to work in cases where the 2D-to-2D matcher fails is the 3D pose estimation and model reprojection step occuring after each 2D similarity matching episode. This series of pose updates incrementally corrects for any foreshortening due to incorrect initial estimates of camera pose. On the other hand, most of the matcher's computation time is spent evaluating the effects of adding or removing pairs of potential correspondences, and in the hybrid matcher, this work is being done using the simpler 2D similarity transformation equations, for which a fast closed-form solution exists. The resulting hybrid system thus offers a clever blend of speed and accuracy.

Figure 10 illustrates the results of the hybrid 3D-to-2D matcher on the same matching problem as shown in Figure 9. Figure 10a shows a zoomed in portion of the initial projection of the 3D site model overlayed onto the image, and Figure 10b shows again the results of the best match found by the 2D-to-2D similarity matching system. The results of the hybrid 3D-to-2D matching system are shown in Figures 10c and 10d. These figures show in turn the results of matching only rooftop lines, and then all wireframe model lines, using a hierarchical matching strategy that will be outlined below. The overlap between the projected model and the 2D image is now very good, and all vestiges of foreshortening errors caused by the incorrect initial estimate of the camera look vector have been removed.

4.1.3 Hierarchical Matching Strategy

Based on experimentation with the model board imagery, further steps have been taken to cut down on the combinatorics of the matching problem. The most relevant of these innovations is to use a hierarchical matching strategy. The run time complexity of correspondence matching in each individual image is governed by the number of model line to data line candidate pairs that are considered. The number of correspondence pairs that are considered is directly related to parameters governing the size of a search window around each model line, and in turn, the size of this search window should be set according to the uncertainty in the user's current knowledge about the pose of the camera. A two stage hierarchical strategy is currently employed. In the initial stage when pose uncertainty is large, only horizontal model lines from building roofs are considered. There are several heuristics behind this choice: roof lines comprise roughly a third of the total number of lines in the full model, they are relatively long (as compared to the building verticals), and they are less likely to be occluded (as compared to the horizontal ground lines).

Once the best corresnondence and pose are found using only roof lines, the second stage of matching begins. The full set of model lines is projected into the image using the computeu pose, and hidden lines are removed. Matching resumes using this more numerous set of lines,

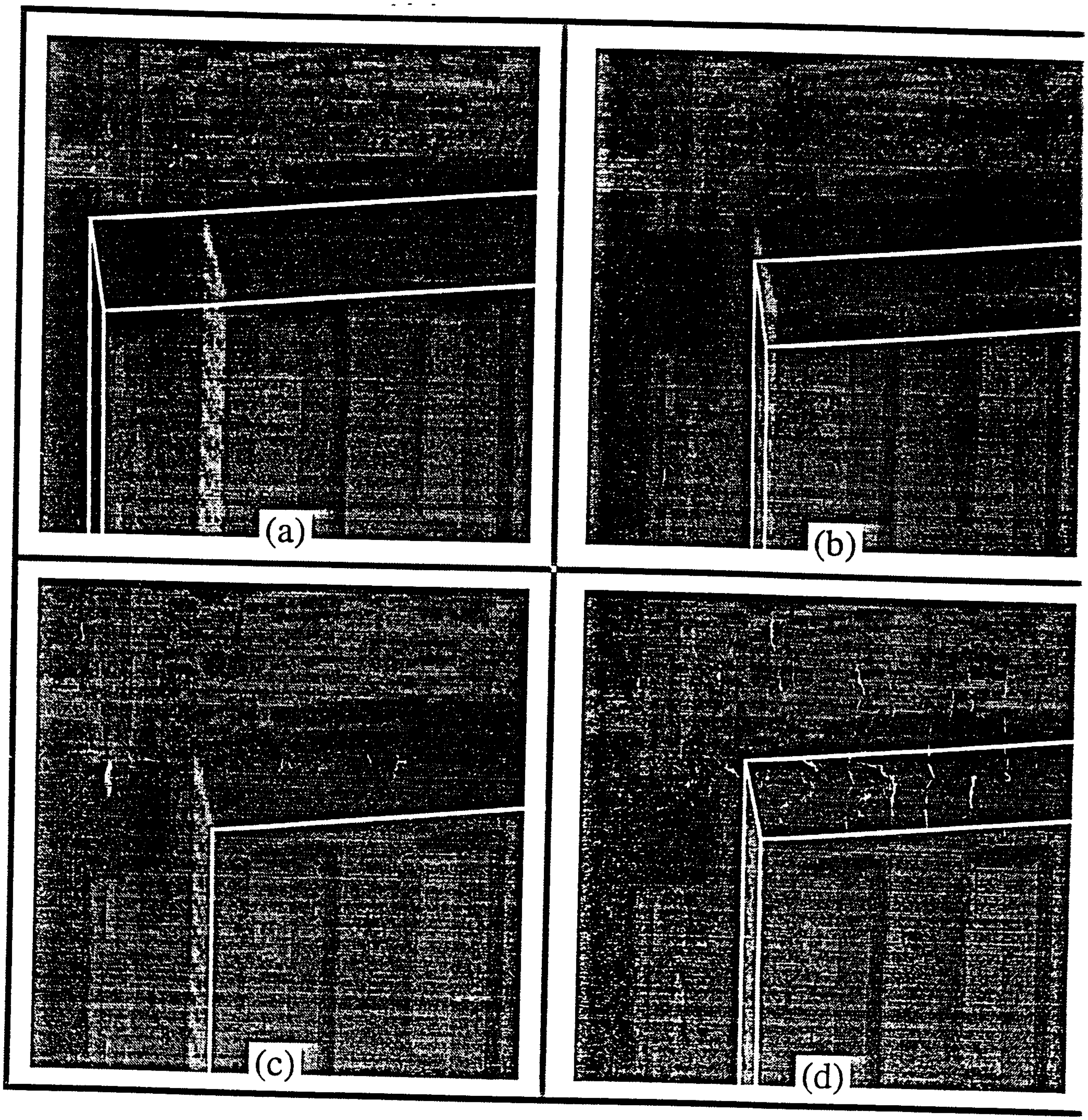


Figure 10: Model matching using the hybrid 3D-to-2D matcher. (a) Close-up of initial projection of model template. (b) Overlay after 2D similarity model matching, showing effects of foreshortening. (c) Close-up of results for 3D-to-2D matching of roof lines. Note that foreshortening effects have been compensated for. (d) Close-up showing alignment after 3D-to-2D matching using the full wireframe model.

but with a much tighter bound on search window size, since the projection of the model into the image using the pose computed from the first stage should be fairly accurate. The goal is to keep a near-constant upper bound on the number of model-to-data correspondence pairs considered at each stage, thereby keeping total match time roughly constant. During the first stage there are fewer model lines, but many more data candidates to consider; during the second stage many more model lines are used, but each has only a small number of data candidates that could possibly be associated with it.

4.2 Camera Resection

The result of model matching is a set of model-to-image feature correspondences between 3D wire-frame edges and 2D image line segments. The next stage in the model extension process uses these correspondences to resect more accurate estimates of camera pose. These updated parameter estimates are then used to triangulate the positions of new scene features. Since automatically generated feature correspondences may contain gross errors, called *outliers*, it is important for any subsequent camera resection procedure to use robust statistical methods when computing new camera parameters. A robust algorithm for finding 3D camera pose from a set of model-to-image point and line-based feature correspondences has been developed at UMass in a Ph.D. thesis by Kumar [24, 25, 26].

At the heart of Kumar's robust pose code is an iterative, weighted least-squares algorithm for computing pose from a set of correspondences that are free from outliers. The pose parameters are found by minimizing an objective function that measures how closely projected model features fall to their corresponding image features. For each point correspondence the objective function is incremented by the squared residual distance from the image point to its corresponding projected model point, weighted by the covariance of the extracted image point. For each line correspondence, the objective function is incremented by the sum of squares of perpendicular residual distances from the endpoints of the image line to the projection of its corresponding, infinitely-extended model line, weighted by the expected covariance in the measured image line endpoints. Although systems in the past have proposed similar objective functions, Kumar's method of solving for the pose parameters differs from previous approaches in two significant ways. First, both rotation and translation parameters are solved simultaneously, which makes more effective use of the geometric constraints and is more accurate in the presence of noise than techniques that decompose the problem by solving for rotation first, followed by translation. Second, the nonlinear least-squares optimization algorithm used to solve for rotation and translation is based on the quaternion representation of rotations, which provides much better convergence properties than solution methods based on Euler angles. The results of this basic pose algorithm are a set of pose parameters that minimize the objective function and a covariance matrix that estimates the accuracy of the solution.

It is well known that least squares optimization techniques can fail catastrophically when outliers are present in the data. For this reason, Kumar embedded the basic pose algorithm described above inside a least median squares (LMS) procedure that repeatedly samples subsets of correspondences to find one devoid of outliers. This approach is called least median squares because it in effect minimizes the median-squared residual distance error rather than the mean-squared distance. LMS is robust over data sets containing up to 50% outliers.

The LMS algorithm works by repeatedly choosing subsets of a given size K from the full set of available correspondences, and computing their implied pose using the basic pose algorithm. In experiments, a subset size of K=6 has worked well [25]. For each subset, the residual squared distance term associated with each correspondence in the set is ranked according to magnitude, and the median residual is noted. After all subsets of size K are processed, the one yielding the smallest median residual error is selected as being outlier-free. Since processing all subsets of size K can be computationally expensive when there are a large number of correspondences, in practice, just a random sample of these subsets is generated, where the number of random subsets that need to be considered is determined on probabilistic grounds based on an estimate of the probable number of outliers present in the data.

After finding a (presumably) outlier-free subset, its computed pose could be used as a robust pose estimate of the full set of correspondences. However, one final stage of processing helps to improve the statistical properties of the pose solution. First, the full set of original correspondences is examined using the pose computed for the best subset, and correspondence pairs are removed that have a large residual error as compared with a threshold determined by robust estimation of the mean and variance of the residual errors. Then the weighted least-squares pose solution is run on the remaining correspondences, which are assumed to be outlier-free. This last computed pose is returned as the final pose estimate. The reason this last stage is performed is to increase the relative statistical efficiency (decrease the variance) of the final estimated pose parameters.

Based on a model of image noise and the assumption that the 3D model data is accurate, closed form expressions for the uncertainty in the pose estimation results have been derived. Kumar has shown analytically that the error in the output pose parameters is linearly related to the noise in the input feature data [25]. He also studied the effect of errors in estimates of the camera principle point and focal length on the resulting pose, showing that incorrect knowledge of the principle point does not significantly affect the computed 3D location of the sensor (although the computed rotation is affected), and that incorrect estimation of the camera focal length only significantly affects the estimate of camera distance from the scene.

For images where internal camera parameters are not available, or not known very accurately, the pose determination process could conceivably be extended to solve for both lens (internal orientation) *and* pose (external orientation) parameters. The resulting highly nonlinear set of equations could best be solved if multiple images taken with the same camera were available, in which case, a joint optimization procedure could be used to determine the single set of lens parameters at the same time that the different pose parameters for each view were being computed. We are not actively investigating this approach since we have been assured that a well-calibrated set of lens parameters will always be known in advance.

4.3 Evaluation of Matching and Pose

The Martin Marietta, Denver Colorado stereo pair distributed by TEC provided a nearly ideal set of data for quantitatively evaluating the procedural chain of line extraction, model matching, and pose determination. Previous evaluation of the accuracy of pose determination using the model board imagery had been hampered by the lack of good ground truth pose measurements. In contrast, very good ground truth camera poses were provided with the Denver stereo pair. The only drawback is that no ground truth model of the building was provided, and no 3D control points were available from which such a model could be constructed. Since model matching and pose determination require an initial 3D model, one was built by hand from the stereo pair by hand-matching distinctive corner points between the two images and then triangulating to derive their 3D positions. From these points, an initial building model composed of 3D line segments was created.

The camera model provided by TEC with the Denver images was more elaborate than the one used by our model matching and pose determination algorithms. This prompted us to update our camera model to a pinhole perspective projection followed by a full six parameter affine lens transformation. This is the most general linear camera model used in computer vision, and was adequate for representing the imaging transformations involved in the Denver stereo pair. Our camera model still does not take into account nonlinear lens aberrations, however no significant amount of nonlinear distortion was present in these images.

Figure 11 shows the results of feature extraction, model matching and pose determination for one of the two stereo images, labeled 75nxx. Figure 11a shows a set of line segments extracted from a portion of the image using the Burns straight line extraction algorithm [11]. The initial 3D model lines built by hand are shown in Figure 11b. The hybrid 3D-to-2D model matching algorithm [9] was used to automatically register these 3D model lines to the set of extracted 2D image line segments. Figures 11c-d show close-ups of the resulting model-to-image registration. Following model matching, an estimate of the optimal camera pose for bringing the model into registration with the image data was computed via a robust pose determination algorithm [25]. This computed pose was then compared to the ground truth pose provided by TEC. For image 74SXX, the difference between computed and ground truth camera locations was 8.73 meters, while for image 75NXX, the difference was 6.91 meters. The average ground truth distance from the camera to the building was 831 meters for image 74SXX and 821 meters for image 75NXX; therefore the relative error between estimated and ground truth camera locations with respect to distance from the modeled object was 1.05% and 0.84% respectively. Angular deviation between computed and ground truth camera look vectors was also computed, and found to be 0.54 degrees for image 74SXX and 0.41 degrees for image 75NXX. In both cases, the agreement between estimated camera pose and ground truth pose is remarkably good, considering the limited extent in the image of the building being used in the model matching and pose determination process.

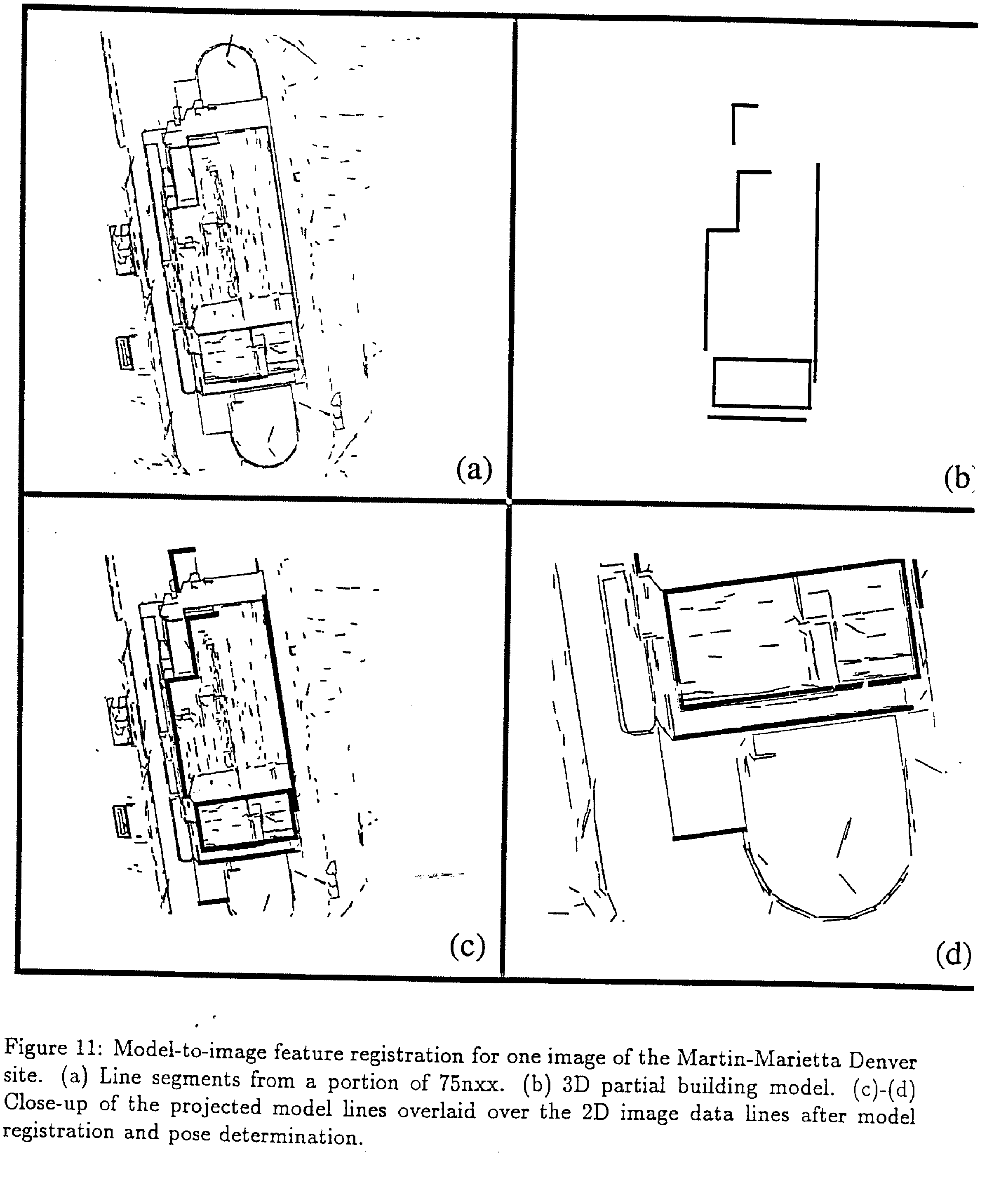


Figure 11: Model-to-image feature registration for one image of the Martin-Marietta Denver site. (a) Line segments from a portion of 75nxx. (b) 3D partial building model. (c)-(d) Close-up of the projected model lines overlaid over the 2D image data lines after model registration and pose determination.

5 Model Extension

Once a set of images has been registered using a partial site model, the goal of model extension is to find unmodeled structures and add them into the model. This task breaks down into three stages: detecting "interesting" features in each image, finding the correspondence of these features across multiple images, and finally, triangulating to recover 3D feature locations in the local site model coordinate system.

The algorithms in this module are also applicable to site model acquisition and refinement. The process of model refinement updates site feature locations from incoming model-to-image feature matches using the same triangulation algorithm that is used during extension. Model acquisition is basically the invocation of model extension procedures over the whole image. The main difference is that for model extension, the pose of each image can be determined by model-to-image registration using the current partial site model, whereas for model acquisition, the pose must be supplied in some other way.

Figure 12 shows a pictorial sketch of model extension. Figure 12a shows a partial site model that has been aligned to a set of image line segments using the model-to-image registration module. For clarity, only candidate data line segments are shown. Note that one building is conspicuously incomplete; the goal of this model extension example is to complete the building wireframe. In Figure 12b, four corner features were selected (by hand) to be added to the model in order to generate the missing sides of the building roof. The position of these corner features in the image, along with the relative pose between this image and another view, dictates a set of epipolar lines in the second image, along which the corresponding corner features must lie. These lines can be truncated at maximum and minimum disparity bounds based on global site model height bounds to determine a set of epipolar line segments, shown in Figure 12c. The search for corresponding corners in the second image is carried out along these bounded segments. From the corresponding image corners that are found, the locations of 3D corner features are triangulated, and the missing roof edges are then added to the model. The extended model can now be used for other site model tasks. Figure 12d shows a close-up of the new site model registered to a third image, focusing on the line segments that were added to complete the partial building model. Note that the position and orientation of the new lines coincide well with the underlying image.

The remainder of this section describes algorithms for epipolar feature matching and triangulation, and considers the limitations of model extension based on low-level features such as points and lines. The use of structured object descriptions is proposed as a way around these limitations.

5.1 Epipolar Matching

Epipolar matching is easiest to describe for point features. To find the 3D location in the scene of a selected image corner point via triangulation, the corresponding projection of that corner feature in a second image must be located. From a single image, all that can be determined is that the 3D corner point lies along a viewing ray in the scene, which can be reconstructed from the 2D image corner location and the image acquisition parameters. This

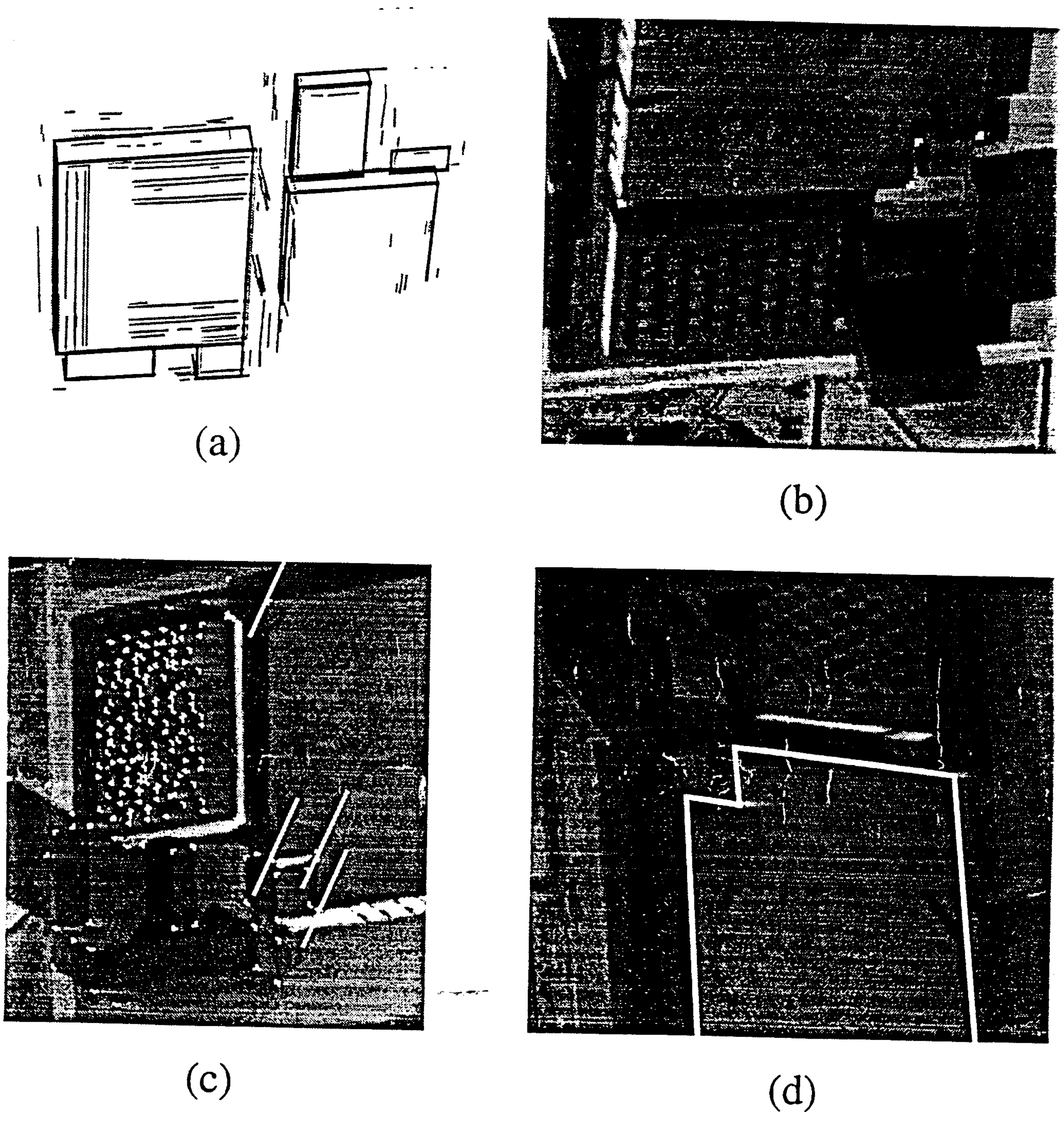


Figure 12: A pictorial summary of 3D model extension. (a) Partial site model registered with an image. (b) Corner points to be used to complete one building wireframe. (c) Epipolar line segments overlayed onto corner points in a second image for epipolar feature matching. Matched corners are then triangulated. (d) The completed building model, automatically registered with a third image (close-up of the wireframe edges that were added).

ray, as seen from a second camera, appears as an *epipolar line* of points in the second image, and the corresponding projection of the 3D point into the second image must lie on this line. This is the well-known *epipolar constraint*, a good discussion of which appears in [5].

The epipolar constraint greatly simplifies the search for corresponding features in the second image by cutting the size of the search space down from the whole image to a thin region straddling the appropriate epipolar line. One additional simplification is possible when the maximum and minimum height of features in the scene can be roughly bounded. In this case, the infinite viewing ray of possible 3D point positions can be truncated at the maximum and minimum Z-coordinate height bounds, and the resulting 3D line segment projected to form a 2D *epipolar line segment* in the second image. This strategy cuts down the size of the search region even more, leading to faster and more reliable matching results.

The main problem in epipolar matching is ambiguity. When many corner features have been detected in the image, it is highly likely that several of them will lie in the epipolar search region. In this case, the matching problem boils down to disambiguating among the many potential matches to settle on a single best one. Our current implementation uses a very simplistic strategy, namely filtering corner features on expected orientation and then choosing the one that lies closest to the epipolar line. More sophisticated matching strategies are possible, the most promising being the use of a third image for disambiguation [5]. The idea is to perform epipolar matching from image 1 to image 2, to get a set of potential matches, then from image 1 to image 3 to get another set of possibilities. These sets of potential matches are then filtered for consistency with respect to each other; each pair of disparities (1-2) and (1-3) must be consistent with coming from the same 3D point in the scene. Many are not, and are weeded out as potential matches. We are currently evaluating this approach, and are also generalizing the epipolar matching system to handle line segment features.

5.2 Triangulation

Using the computed model-to-image feature matches determined via epipolar matching, multi-image triangulation is performed to locate new 3D model points and lines in the model coordinate system. Currently, only code for triangulation of point features is implemented. The estimation of new 3D points can be done in either batch or iterative sequential mode. Triangulation requires at least two frames, therefore the minimum batch size is two. Results from different batches can be be integrated by the standard Kalman-filter covariance-based updating equations.

Due to noise both in image measurements and camera pose estimates, image projection rays will not exactly intersect at a point. Kumar has developed a 3D pseudo-intersection method that minimizes an error equation based on the same constraints that determine the pose [25]. The criterion underlying this error equation is that the best estimate for any model point location is the point that minimizes the least-squares distance between the predicted image location of the projected model point and its actual image location, taking into account covariances in the measured image positions and the computed pose. Two non-linear error equations are obtained for each scene point for each image frame, thus a minimum of two frames is needed to solve the system of equations. Techniques for the solution of nonlinear systems of equations generally require an initial estimate that is close to the true solution. The initial estimate in this case is chosen as the point that minimizes the sum of squares of perpendicular distances to all the image projection rays, a point that is easily found by solving a linear system of equations. Using this initial guess, an iterative procedure is employed to solve the system of non-linear equations for each point. The iterative procedure is repeated until there is convergence. Usually only one iteration is sufficient for accurate results [25]. A byproduct of this calculation is an approximate covariance matrix for the derived 3D model point position.

The method described above can also be used for model refinement. In this case initial model points have input covariances associated with them. The pseudo-intersection method is used to calculate a new estimate for each initial model point. The covariance matrices of a new estimate and an initial model point are used to fuse the two estimates and provide a new uncertainty matrix using the standard Kalman filtering equations.

Kumar assumes that the lens parameters are known, so only uncertainty in the pose parameter estimates is considered when computing the error in a triangulated point position. We are extending this approach to handle uncertainty in the camera lens parameters as well. We are also extending the triangulation equations to work for lines as well as points.

5.3 Structured Object Recovery

Model extension work at UMass previous to the RADIUS project was based on tracking point and line features through a motion sequence of images [26]. This is clearly inappropriate for the present application where images from widely separated viewpoints are used. Instead, an algorithm to perform feature matching along epipolar lines was programmed and applied to the task of determining corresponding corner features across a pair of images. The initial results were not encouraging: due to the large number of detected corner features in each image and the imprecision in the computed epipolar geometry, many ambiguous potential matches were identified for each corner. This is a common problem, and a number of solutions have been proposed in the literature, including filtering inappropriate image matches based on 3D constraints and on epipolar lines from 3 or more views.

Even if correct epipolar matching of individual point and line features is achieved, subsequent triangulation using these features yields little more than a cloud of 3D corners and lines. Further processing is still needed to merge these primitive 3D features into structured objects of interest (such as buildings). Because of these considerations, we are now exploring an alternative approach based on more structured sets of image features, such as connected components of lines and corners that form a partial or complete building wire frame. A project is now underway to perform perceptual grouping of lines and corners into polygons and wire frame structures, match them across multiple images using the precomputed epipolar geometry, then finally triangulate these structures using multiple images and 3D geometric object constraints such as incidence, parallelism, perpendicularity and coplanarity.

6 Image-to-Image Registration

Image-to-image registration involves determining feature correspondences between two or more images of the same scene, along with computation of the relative pose between them. Clearly, if a model of the scene is available, model-to-image registration (Section 4) could be applied to each image separately, and then the relative pose between images could be computed from each image's absolute pose. However, prior to initial model acquisition, this is not an option. If the pose for each image is already known, then features can be matched between images using an epipolar matching algorithm as described in Section 5, even when no 3D model is available. But when presented with a new set of images of an unmodeled site, neither of these approaches is applicable and another more general technique must be used. Fast and reliable general image-to-image matching techniques exist when the distance between views is small [2] or when features can be tracked through a motion sequence [26]. What is lacking are good methods for finding matches in monocular images taken from arbitrary viewpoints, as is the case in the RADIUS domain.

In this section a simple method is presented that allows fast and accurate matching of coplanar structures across multiple images. We show that the full perspective matching problem for horizontal coplanar structures can be reduced to a simpler four parameter affine (similarity) matching problem when the horizon line of the scene can be determined in the image. Given the horizon line, the image can be transformed to show how the scene would appear if the camera's principle axis was directed straight down, parallel to gravity. That is, given the horizon line in an aerial photograph, the image can be artificially warped to produce a *nadir view* of the same scene. This process is called *rectification* in aerial photogrammetry.

6.1 Nadir Views

Nadir views are very popular in aerial image understanding systems because they simplify the description of the imaging process; for example, flat, rectangular building rooftops appear as rectangles in the image. Constraining the camera orientation to take a nadir view fixes two degrees of its rotational freedom. The four remaining degrees of freedom, one free camera rotation about the principle axis and three translation parameters, can be characterized by how they affect the appearance of a horizontal plane (such as a building roof) in the image. For example, translation directly towards or away from the object plane manifests itself as a uniform change of scale in the projected image. Translation parallel to the planar surface shows up as a proportional 2D translation in the image. Finally, a rotation of the camera about its principle axis causes the projected image to rotate by the same angle about a point in the image plane. The projected image of a horizontal plane in a nadir view is therefore described by four affine parameters that are directly related to the physical pose of the camera. In other words, the function that maps object coordinates to image coordinates for a horizontal planar structure in a nadir view is a four parameter affine mapping, often called a *similarity mapping.*

For oblique views, the function that maps horizontal object coordinates into image coordinates is no longer a similarity mapping, but is instead, a more general projective transformation [17]. Parallel roof lines then appear to converge in the image plane, intersecting at a *vanishing point.* Two or more vanishing points from different sets of horizontal, parallel lines, form a line in the image called the *vanishing line* or *horizon line.* In nadir views, all parallel scene horizontals remain parallel in the image, and the horizon line is said to be "at infinity". This convention allows the relation between oblique and nadir views to be precisely stated. Parallel horizontal lines in the scene map to lines in the image that are directed towards a vanishing point that lies on the horizon line. Nadir views are distinguished from oblique views in that the horizon line is located at infinity. In this case, the projective transformation mapping object coordinates from a horizontal plane into image coordinates is a similarity transform.

These considerations lead to a simple yet powerful observation. By applying a projective transformation that maps the horizon line in the image to the line at infinity, the vanishing points of all horizontal lines will also appear at infinity. After this mapping has been performed, all parallel horizontal lines in scene will appear parallel in the image. This implies that the new image is a nadir view of the scene, and thus, the projective mapping of scene horizontals into the image can be represented as a similarity transform.

6.2 Rectification

To take advantage of the simplicity of nadir views for high altitude photographs, we designed a two stage approach to matching planar structures viewed from any orientation. The approach consists of applying information-preserving transformations to the image to reduce the complexity of the matching problem. First, vanishing point analysis is performed to locate the horizon line in the image (see Section 3), and a projective transformation is applied that rectifies the image. Second, the scene is assumed to be approximately planar with respect to camera altitude, and image features are matched using a 2D similarity transformation version of the local search matcher described in Section 4.

For a pinhole camera image, the location and orientation of the horizon line determines the 3D orientation of the camera with respect to gravity. When the equation of the horizon line is $ax + by + c = 0$, the gravity vector, in camera coordinates, is

$$n = (a, b, c)/||(a, b, c)||. \tag{3}$$

For a nadir view, the gravity vector must be parallel to the Z-axis of the camera. If the camera could move, an oblique view could be changed to a nadir view by merely rotating the camera to point straight down. The camera can no longer be moved physically, of course, but the image *can* be transformed artifically to achieve the desired 3D rotation.

Assume the unit orientation of the gravity vector has been determined to be n, as in equation 3, oriented into the image ($c \geq 0$). To bring this vector into coincidence with the positive camera Z axis requires a rotation of angle $\mathrm{Cos}^{-1}(n \cdot (0, 0, 1))$ about the axis $n \times (0, 0, 1)$. The effects of this camera rotation on the image can be simulated by an invertible projective transformation in the image plane [23]. In homogeneous coordinates,

$$k_i \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \begin{bmatrix} E & F & a \\ F & G & b \\ -a & -b & c \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

where

$$E = \frac{a^2c + b^2}{a^2 + b^2}, F = \frac{ab(c-1)}{a^2 + b^2}, G = \frac{a^2 + b^2c}{a^2 + b^2}.$$

The image is transformed to appear as if the camera had been pointing straight down, parallel to gravity. The result, therefore, is a rectified nadir view of the scene.

6.3 Example

Because it relies on 2D image properties, image rectification is applicable when no prior scene model is available. It thus lends itself well to the problem of image-to-image matching of horizontal coplanar structures. In this case, both images are rectified using vanishing point information, and one is treated as the model while the other becomes the data to be matched. The goal is then to discover the similarity transformation that maps one set of rectified image lines into another. We use the local search matching system described in Section 4. Although aerial images do not often depict purely planar scenes, the method can still be used to get coarse matching results in cases such as high-altitude photos, or for mapping road networks on a nearly flat ground plane.

Figure 13 shows an example of rectification-based aerial image registration. Figures 13a and 13b show sets of straight line segments extracted from two aerial photographs of Fort Hood, Texas. The first image presents a nadir view of the scene, a fact verified by vanishing point analysis, which finds two orthogonal sets of nearly parallel lines. The second image is clearly not a nadir view, a fact again verified via vanishing point analysis. Figure 13c shows these image lines after applying the automated image rectification procedure described above. To unwarp the second image a generic camera model was assumed, placing the principle point in the center of the image and optical axes along the row and column axes of the raster image. The focal length was computed by finding the distance of the focal point from the image that resulted in perpendicularity of the two vectors from the (variable) focal point towards the two (fixed) vanishing points in the image. The aspect ratio was assumed to be one-to-one.

To apply local search matching, image 1 was assumed to be the model, and rectified lines from image 2, the data. Both line sets were filtered to include only lines greater than 100 pixels long, reducing the matching problem to 55 long lines in one image and 68 lines in the other. Additionally, the search space was partitioned based upon the dominant orthogonal line directions. The best match found is displayed in Figure 13d.

6.4 Limitations

Although automated image rectification based on vanishing point analysis works well in urban and industrial aerial image domains, the combination of rectification followed by

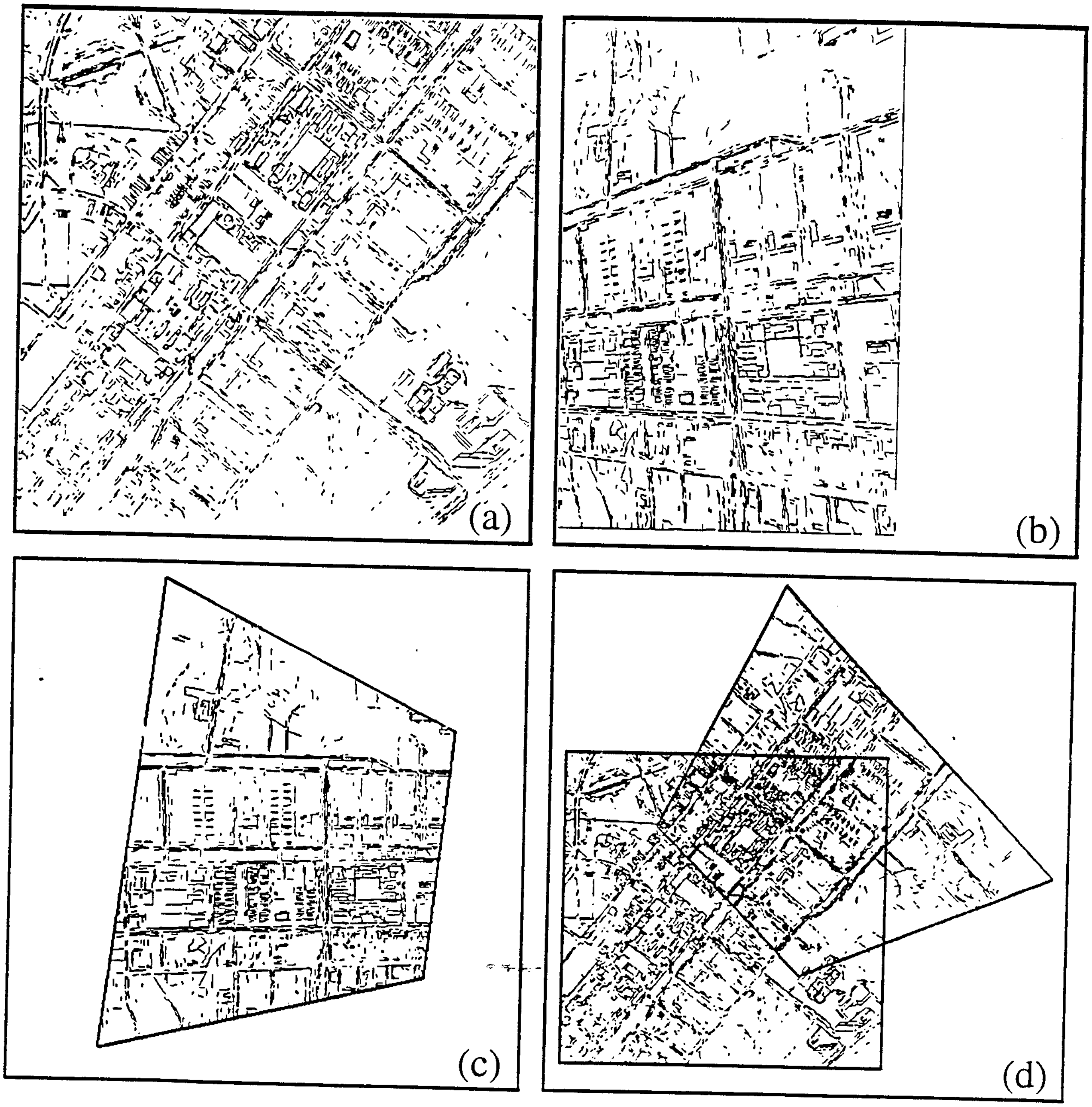


Figure 13: Image-to-image matching example on an aerial image: (a) image lines from nadir view, (b) image lines from oblique view, (c) rectified oblique view, (d) registration of nadir view with rectified oblique view.

similarity matching to determine image-to-image feature correspondences is limited. The approach works well for relatively high-altitude photographs, like the Fort Hood, Texas images shown here, where both ground-level structures and building rooftops can be treated as nearly coplanar. The approach does not work well on the model board images, however, since these images were taken at a lower altitude where the 3D structure of the buildings becomes very apparent. As images in the RADIUS domain are expected to be more like the model board images than the Fort Hood ones, the method for image-to-image matching described in this section is no longer being pursued.

However, to the extent that *some* scene features are found to be coplanar and can be successfully matched, this initial set of planar correspondences provides strong constraints on the positions of remaining features. For calibrated cameras, the relative rotation and direction of translation between two camera positions can be computed from the perspective transformation describing how the appearance of a planar structure differs in the two images [17]. This reduces the search for other 3D feature correspondences to that of induced stereo, where corresponding feature points lie along known *epipolar lines* (see Section 5).

7 Projective Structure Recovery

In addition to traditional metric approaches to scene reconstruction via pose determination and triangulation, we are pursuing a complementary research path that investigates the use of projective invariants for image registration, image transference, and scene reconstruction. The benefit of this approach is that dependence on initial estimates of the camera pose parameters are minimized. Experimentally, we have noticed that techniques based on generalizing 2D planar invariants to account for non-coplanarity seem more robust than those based on full 3D to 2D invariants (e.g. methods based on the essential matrix). We are working under the hypothesis that this is so because the scene features projected onto an aerial photograph are nearly coplanar with respect to distance from the camera. Full 3D approaches work best for sets of point positions that vary significantly in all three dimensions, but often fail when the points are coplanar. One should not, therefore, expect such methods to work well on aerial images.

7.1 Invariant-Based Model Extension

Previous work by Collins investigated invariant recovery of planar scene features [13]. Given knowledge of at least four coplanar points or lines in the scene and their correspondence in an image, a planar projective transformation (called a homography) that maps new points and lines from the image into their proper locations on the object plane can be estimated, without first computing pose or calibrating the camera.

When other, non-coplanar scene features are present, the mapping from world to image is no longer completely described by a homography, and planar model extension is no longer applicable. We have begun to explore approaches for extending planar model extension to handle more general 3D scenes. Assume that two views of a scene are available, that at least four coplanar points or lines are available for use as a reference plane, and that an image-to-image homography has been computed that transforms the projections of reference plane point features from image one into their corresponding projections in image two. What can be said about the transformed image of a scene point outside of the reference plane? The location predicted for it by the planar homography will not, in general, coincide with the actual location of the projected scene point in image two. However, the residual difference between the predicted and actual positions of the projected point in image two are constrained to lie along lines intersecting in a single point. These lines are called epipolar lines, and they intersect at the epipole, which geometrically corresponds to the image of the focal point of the camera when image one was taken. Furthermore, the direction of each residual difference vector, either towards or away from the epipole, determines whether the corresponding 3D scene point lies either forward or behind the plane of reference. These results are valid regardless of either camera's location, orientation, or calibration parameters.

For illustration, two aerial photographs are shown in Figures 14a and 14b. Figures 14c and 14d show thirty seven corresponding pairs of points that were chosen by hand from these images. Several sets of points delimit the tops of buildings, and are therefore coplanar in the scene. The four pairs of coplanar points marked with a cross delineate the corners

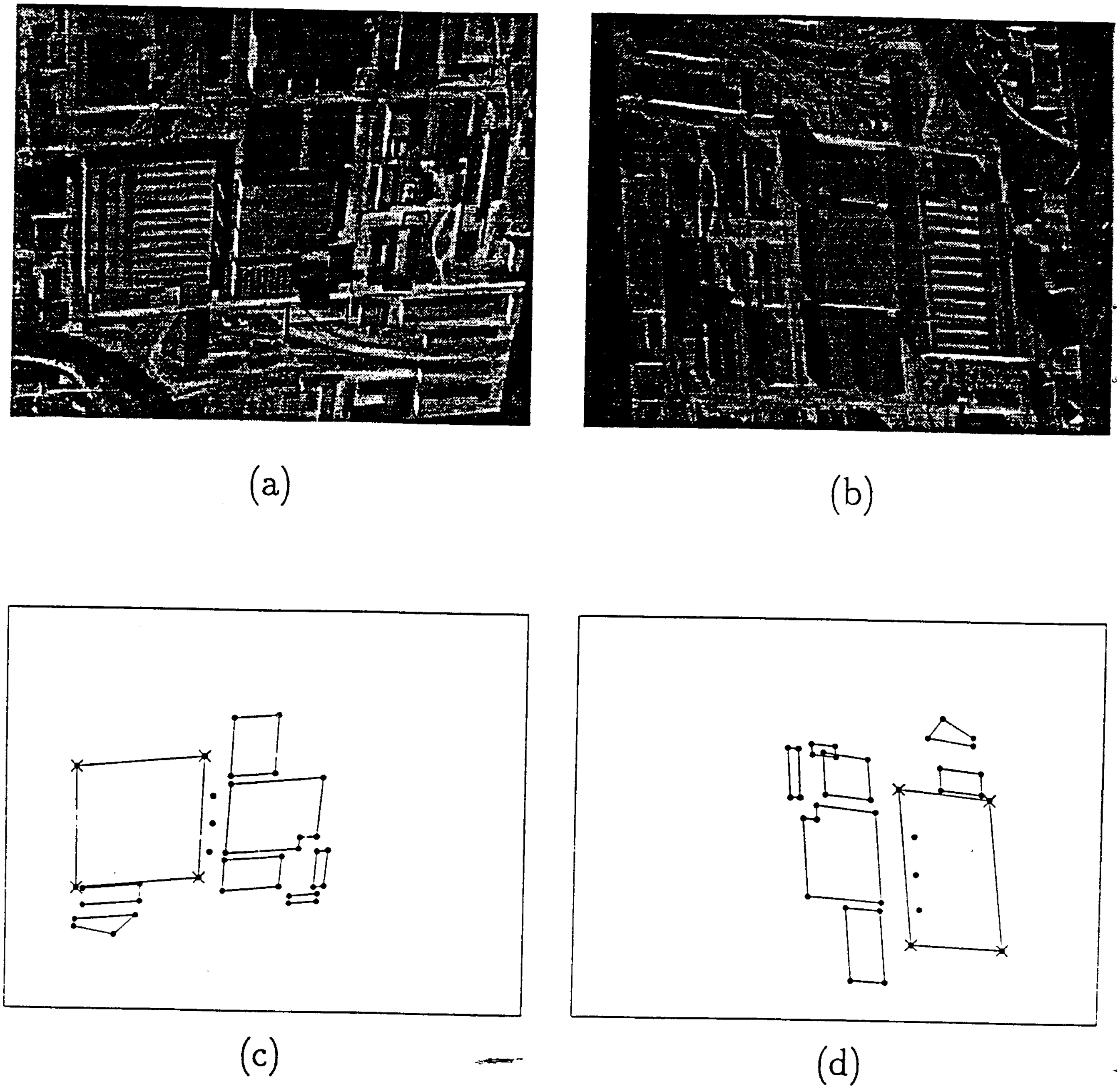


Figure 1: Two aerial photographs that are not well-approximated by a single plane. (a) RADIUS Model Board 1, Image J8. (b) RADIUS Model Board 1, Image J2. (c) Interesting points extracted by hand from Image J8. (d) Corresponding points extracted by hand from Image J2. Some building boundaries have been added for clarity. Crosses mark points that will be used to estimate a homography between the two images.

of a rooftop, and these were used to estimate a homography from the first image into the second. All points from the first image .were then mapped into the second image using this homography, and their positions were noted. Figure 15 shows residual difference vectors between predicted locations (in black) of transformed points from image one, and actual point locations (in white) where they were found in image two. The four pairs of coplanar reference points align exactly, as they must by definition of the homography.

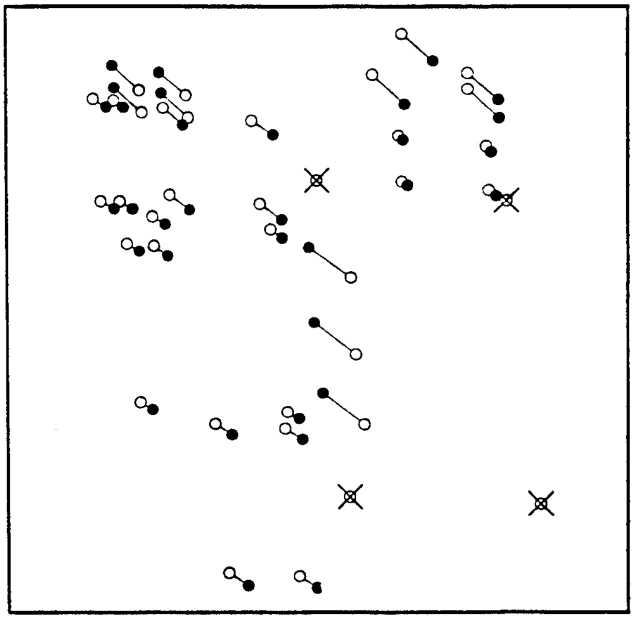


Figure 15: Difference vectors between points from image one transformed by a planar homography into image two (black dots), and their corresponding actual positions in image two (white dots). Points marked with crosses were used to define the homography.

All remaining residual vectors lie along infinite epipolar lines that intersect at a single epipolar point, which in this case, is far off the image. Furthermore, note that the difference vectors for structures taller than the rooftop used to compute the homography are oriented in one direction, while difference vectors for structures shorter than the rooftop are oriented in the opposite direction. This property holds in general, and can be used to qualitatively partition scene points into three categories depending on the orientation of their residuals: those lying closer to the viewer than the reference plane, those lying on the plane (difference vector is zero), and those lying further away.

7.2 Approximate Height Recovery

A more detailed analysis determines the extent to which quantitative information can be recovered from this geometric configuration. This analysis is published in [14], and will not be repeated here. The results of the analysis show that when the epipole is at infinity (and thus the residual difference vectors are parallel in the image) the length of each residual vector is directly proportional to the perpendicular distance of the corresponding 3D scene point from the reference plane. If the reference plane has been chosen as some horizontal plane, such as the top of a building or the ground plane, then the relative heights of each scene point can be determined. When the epipole is not exactly at infinity, but is still far away from the center of the image, then the relative heights of scene points are recovered approximately. For a horizontal reference plane, situations leading to nearly parallel residual difference vectors include sequences of images taken from camera positions at roughly the same altitude. Because this method is based on projective invariants, the exact altitude, orientation, and intrinsic calibration parameters of the camera (or cameras) do not need to be known.

An initial test into the feasibility of this approach to height recovery was conducted using model board images J2 and J8. The four corners points of building 34 (the one with many long, parallel roof vents) were identified in both images and the projective transformation relating the two sets of points was computed. This transformation was used to map several building corner points from image J2 into image J8, and the lengths of the residual difference vectors between their transformed positions and actual observed locations in image J8 were measured. As described above, these lengths should be approximately proportional to the heights of each point with respect to the height of building 34. To compare the computed relative heights to known ground truth heights of each point, the single scale factor relating residual vector lengths to model board Z coordinate values was estimated, and the resulting Z values were then converted to absolute height above a nominal ground plane of Z=-2 inches. The average relative error of the computed heights of 23 test points was about 4.8%. The results are shown in Table 2, where for ease of interpretation, all model board units (in inches) have been converted into world units (in feet) using the fact that the model board is a 1:500 inch scale model.

Table 2: Comparison of ground truth point height above the ground plane versus height computed via projective invariants for 23 selected test points on model board 1. See text for details. All heights are reported in feet.

true	29	30	33	35	37	42
computed	29	31	29	28	38	39
rel. error	0%	3%	12%	20%	3%	7%
true	44	46	57	58	59	61
computed	44	43	54	60	60	63
rel. error	0%	7%	5%	3%	2%	3%
true	62	62	64	74	86	86
computed	57	63	63	72	90	96
rel. error	8%	2%	2%	3%	5%	12%
true	86	94	148	179	224	
computed	91	97	145	178	220	
rel. error	6%	3%	2%	1%	2%	

References

[1] Amerinex Artificial Intelligence. *The KBVision System User's Guide.* Amerinex Artificial Intelligence, Amherst, MA, 1991.

[2] Anandan, P. "Measuring Visual Motion from Image Sequences," Ph.D. Thesis and COINS Tech Report 87-21, University of Massachusetts, Amherst, MA, 1987.

[3] American Society of Photogrammetry. *Manual of Photogrammetry.* Fourth Edition, American Society of Photogrammetry, Falls Church, VA, 1980.

[4] Anderson, T.W. "Estimating Linear Statistical Relationships." *The Annals of Statistics,* Vol. 12(1), 1984, pp. 1–45.

[5] Ayache, N. *Artificial Vision for Mobile Robots.* MIT Press, Cambridge, MA, 1991.

[6] Barnard, S.T. "Interpreting Perspective Images." *AI Journal,* Vol. 21(4), November 1983, pp. 435–462.

[7] Beveridge, J.R., Weiss, R.S. and Riseman, E.M. "Combinatorial Optimization Applied to Variable Scale 2D Model Matching." *Proceedings International Conference on Pattern Recognition*, Atlantic City, NJ, June 1990, pp. 18–23.

[8] Beveridge, J.R. and Riseman, E.M. "Can Too Much Perspective Spoil the View? A Case Study in 2D Affine Versus 3D Perspective Model Matching." *Proceedings ARPA Image Understanding Workshop*, San Diego, CA, 1992, pp. 655–663.

[9] Beveridge, J.R. and Riseman, E.M. "Hybrid Weak-Perspective and Full-Perspective Matching." *Proceedings IEEE Computer Vision and Pattern Recognition*, Champaign, IL, 1992, pp. 432–438.

[10] Boldt, M., Weiss, R.S. and Riseman, E.M. "Token-Based Extraction of Straight Lines." *IEEE Transactions on Systems, Man and Cybernetics,* Vol. 19, No. 6, 1989, pp. 1581–1594.

[11] Burns, J.B., Hanson, A.R. and Riseman, E.M. "Extracting Straight Lines." *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 8(4), July 1986, pp. 425–456.

[12] Collins, R.T. and Weiss, R.S. "Vanishing Point Calculation as a Statistical Inference on the Unit Sphere." *Proceedings Third International Conference on Computer Vision,* Osaka, Japan, December 1990, pp. 400–403.

[13] Collins, R.T. "Single Plane Model Extension using Projective Transformations." *Proceedings ARPA Image Understanding Workshop,* San Diego, CA, January 1992, pp. 917–923.

[14] Collins, R.T. "Projective Reconstruction of Approximately Planar Scenes." in *Interdisciplinary Computer Vision: An Exploration of Diverse Applications,* Jane Harmon, Editor, Proc. SPIE No. 1838, Oct. 1992, pp. 174–185.

[15] Collins, R.T., Hanson, A.R., Riseman, E.M., and Cheng, Y.Q. "Model Matching and Extension for Automated 3D Site Modeling." *ARPA Image Understanding Workshop*, Washington, DC, April 1993, pp. 197–203.

[16] Collins, R.T. and Beveridge, J.R. "Matching Perspective Views of Coplanar Structures using Projective Unwarping and Similarity Matching." *Proceedings IEEE Computer Vision and Pattern Recognition*, New York City, NY, June 1993, pp. 240–245.

[17] Faugeras, O.D. and Lustman, F. "Motion and Structure from Motion in a Piecewise Planar Environment." *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 2, 1988, pp. 485–508.

[18] Fennema, C., Hanson, A.R., Riseman, E.M., Beveridge, J.R. and Kumar, R. "Model-Directed Mobile Robot Navigation." *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 20(6), 1990, pp. 1352-1369.

[19] Gee, S.J. and Newman, A.M. "RADIUS: Automating Image Analysis Through Model-Supported Exploitation." *Proceedings of the ARPA Image Understanding Workshop*, Washington, DC, April 1993, pp. 185–196.

[20] Gerson, D.J. "RADIUS : The Government Viewpoint." *Proceedings of the ARPA Image Understanding Workshop*, San Diego, CA, January 1992, pp. 173–175.

[21] Grimson, W.E.L. and Huttenlocher, D.P. "On the Sensitivity of the Hough Transform for Object Recognition." *Proceedings of the Second International Conference on Computer Vision*, Tampa, FL, 1988, pp. 700–706.

[22] Horn, B.K.P. "Relative Orientation." *International Journal of Computer Vision*, Vol. 4, 1990, pp. 59–78.

[23] Kanatani, K. "Constraints on Length and Angle." *Computer Vision, Graphics, and Image Processing*, Vol. 41, 1988, pp. 28–42.

[24] Kumar, R. and Hanson, A.R. "Application of Pose Determination Techniques to Model Extension and Refinement." *Proceedings ARPA Image Understanding Workshop*, San Diego, CA, January 1992, pp. 727–744.

[25] Kumar, R. *Model Dependent Inference of 3D Information from a Sequence of 2D Images.* Ph.D. Thesis, Computer Science Department, University of Massachusetts, Amherst, MA, February 1992. Also published as CS tech report TR92-04.

[26] Kumar, R., Sawhney, H.S. and Hanson, A.R. "3D Model Acquistion from Monocular Image Seqnences." *Proceedings IEEE Computer Vision and Pattern Recognition*, Champaign, IL, 1992, pp. 209–215.

[27] Liu, Y., Huang, T.S. and Faugeras, O. "Determination of Camera Location from 2D to 3D Line and Point Correspondences." *Proceedings IEEE Computer Vision and Pattern Recognition*, Ann Arbor, MI, 1988, pp. 82–88.

[28] Wang, L.L. and Tsai, W.H. "Camera Calibration by Vanishing Lines for 3-D Computer Vision." *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 13, 1991, pp. 370–376.