THREE-DIMENSIONAL RECONSTRUCTION OF POINTS AND LINES WITH UNKNOWN CORRESPONDENCE ACROSS IMAGES [*]

Y-Q Cheng[*], E.M. Riseman, X.G. Wang, R.T. Collins[**] and A.R. Hanson

Village Networks, Inc
100 Village Court
Hazlet, NJ 07730[*]

Department of Computer Science
University of Massachusetts
Amherst, MA 01003
Email: {riseman}@cs.umass.edu

Robotics Institute, NSH
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213[**]

Abstract

Three-dimensional reconstruction from a set of images is an important and difficult problem in computer vision. In this paper, we address the problem of determining image feature correspondences while simultaneously reconstructing the corresponding 3D features, given the camera poses of disparate monocular views. First, two new affinity measures are presented that capture the degree to which candidate features from different images consistently

[*]This work was funded by the RADIUS project under DARPA/Army contract number TECDACA76-92-R-0028, DARPA/TACOM contract number DAAE07-91-C-R035, DARPA/Army contract number TECDACA76-92-C-0041, and the National Science Foundation (NSF) grant number CDA8922572)

represent the projection of the same 3D point or 3D line. An affinity measure for point features in two different views is defined with respect to their distance from a hypothetical projected 3D pseudo-intersection point. Similarly, an affinity measure for 2D image line segments across three views is defined with respect to a 3D pseudo-intersection line. These affinity measures provide a foundation for determining unknown correspondences using weighted bipartited graphs representing candidate point and line matches across different images. As a result of this graph representation, a standard graph-theoretic algorithm can provide an optimal, simultaneous matching and triangulation of points across two views, and lines across three views. Experimental results on synthetic and real data demonstrate the effectiveness of the approach.

1 Introduction

Three-dimensional model acquisition remains a very active research area in computer vision. One of the key questions is how to reconstruct accurate 3D models from a set of calibrated 2D images via multi-image triangulation. The basic principles involved in 3D model acquisition are feature correspondence determination and triangulation, with the two commonly used types of image features being points and lines. Usually, 2D features are extracted first, such as corners, curvature points, and lines from each image. Then, the correspondence of these features is established between any pair of images, usually referred to as "the correspondence problem". Finally, the 3D structure is triangulated from these 2D correspondences.

Many reconstruction papers assume the correspondence problem has been solved [2, 7, 35, 39, 50, 51, 53]. Unfortunately, in many applications, this information is not available and mechanisms to achieve correspondence are unreliable. This has caused serious criticism of feature-based methods [6, 22, 33, 52, 60, 61]. The process of finding $2D$ image feature correspondences can be computationally expensive and difficult to implement reliably, requiring subsequent algorithms to employ robust mechanisms for detecting outliers due to mismatches [46, 49].

Even if the image feature correspondences are known, robust triangulation of the 3D models using noisy image data is still a non-trivial problem and an on-going research topic. Extensive research has been devoted to developing robust algorithms in this area [5, 8, 23, 37, 46, 49, 66, 67], including processing of monocular motion sequences, stereo pairs, and sets of distinct views. Although both point-based and line-based triangulation are commonly employed, more attention has been paid to line-based triangulation since it generally provides more accurate reconstructions.

In this paper, we address the problem of determining image feature correspondences given known camera poses, while simultaneously computing the corresponding 3D features. We restrict our attention to simultaneous determination of image feature correspondences and recovery of their 3D structure using matching and triangulation of noisy 2D image points and lines. Our approach assumes a set of calibrated images, for which both intrinsic (lens) parameters and either absolute or relative poses are known. Therefore these approaches are well-suited for photogrammetric mapping applications where extrinsic parameters are already known, such as 3D aerial reconstruction in cultural settings [4], for wide-baseline multi-camera stereo systems, or for model extension applications where a previous partial model has been used to determine camera pose for a set of new views, from which previously unmodeled scene features are now to be recovered [18, 19, 21, 46].

This paper is organized as follows. Section 2 reviews previous related work in the area of 3D reconstruction with unknown apriori correspondences. Section 3 introduces two new affinity measures for determining image point and line correspondences across images, and uses them to construct weighted bipartite graphs. Section 4 formulates the image feature matching problem as the general maximum-weight bipartite matching problem and develops two algorithms to simultaneously match and reconstruct 3D points and lines from noisy 2D image points and lines, respectively. Finally, Section 5 and Section 6 present and analyze experimental results from synthetic and real image data sets. Section 7 gives our conclusions.

2 Previous Work

2.1 Motion Estimation without Correspondences

Aggarwal *et al* [2] reviewed the correspondence problem two decades ago. In recent years, a variety of correspondence problems [2, 6, 13, 14, 16, 33, 39, 51, 52, 61, 63] have been studied. In addition, many researchers have worked on the problem of motion estimation without pre-specified correspondences [3, 22, 33, 39, 50, 52, 63].

Aloimonos, *et al* [3], presented an algorithm to estimate 3D motion without apriori correspondences by combining motion and stereo matching. Huang and his research group [33, 50, 52] presented a series of algorithms to estimate rigid-body motion from $3D$ data without matching point correspondences. Goldgof *et al* [33] presented moment-based algorithms for matching and motion estimation of 3D point or line sets without correspondences and applied these algorithms to object tracking over the image sequences. The basic idea is to find two coordinate systems based on relative positions of 3D points/lines before and after the motion, then compute the motion parameters (rotation and translation) that make these coordinate systems coincide. The disadvantages of the approach include sensitivity to noise and to missing or false points or lines.

Lee *et al* [52] proposed an algorithm to deal with the correspondence problem in image sequence analysis. This method is based on the following three assumptions: (1) the objects undergo a rigid motion; (2) a perspective projection camera model can be used; (3) the translation vector is small compared to the distance of the object.

Recently, we presented a mathematical symmetry in the solutions of rotation parameters and point correspondences, derived a closed-form solution based on eigenstructure decomposition for correspondence recovery in ideal cases with no missing points, and developed a weighted bipartite matching algorithm to determine the correspondences in general cases where missing points occur [63]

2.2 Determination of Correspondences from Nonrigid Objects

Objects in the world can be nonrigid, and an object's appearance can deform as the viewing geometry changes. Consequently, research has been carried out to address the problem of correspondence and description using deformable models [56, 57, 59, 60, 61].

Scott and Longuet-Higgins [60] developed an algorithm to determine the possible correspondences of 2D point features across a pair of images without use of any other information (in particular, they had no information about the poses of the cameras). They first incorporated a proximity matrix description which describes Gaussian-weighted distances between features (based on inter-element distances) and a competition scheme allowing candidate features to contest for best matches. Then they used the eigenvectors of this matrix to determine correspondences between two sets of feature points.

Shapiro and Brady [61] also proposed an eigenvector approach to determining point-feature correspondence based on a modal shape description. Recently, Sclaroff and Pentland [59] described a modal framework for correspondence and description. They first developed a finite element formulation using Gaussian basis functions as Galerkin interpolants, then used these interpolants to build stiffness and mass matrices. Correspondences were determined by decomposing the stiffness and mass matrices into a set of eigenvectors.

2.3 Determination of Correspondences among Disparate, Monocular Images

Methods based on tracking features such as points and line segments through a sequence of closely-spaced image frames cannot be applied in our present domain, since they are based on small-motion approximations, while we are presented with a set of discrete, disparate, monocular views. Furthermore, heuristic measures based on similarity of image feature appearance across multiple images will also fail, since widely disparate viewpoints, taken at different times of day and under different weather conditions can lead to corresponding image features of significantly different appearance. Gruen and Baltsavias [35] describe a constrained multi-image matching system where intensity templates extracted from one reference image are affine-warped and correlated along epipolar lines in each other image. Kumar *et al* [48] present a multi-image plane+parallax matching approach where they compensate for the appearance of a known 3D surface between a reference view and each other view, then search for corresponding points along lines of residual parallax.

Collins [16] introduces the term "true multi-image" matching and present a new space-sweep approach to multi-image matching that make full and efficient use of the geometric relationships between multiple images and the scene to simultaneously determine 2D feature correspondences and the 3D positions of feature points in the scene.

In their recent work, Bedekar and Haralick [7] first pose the triangulation problem as that of finding the Bayesian maximum a posteriori estimate of the 3D point, given its projections in N images, assuming a Gaussian error model for the image point coordinates and the camera parameters. Then, they consider the correspondence problem as a statistical hypothesis verification problem and solve this problem by an iterative steepest descent method.

Recently, graph theoretic methods have been applied [13, 14, 30, 34, 43, 58, 63, 64, 65]. Gold and Rangarajan [30] presented a new algorithm for graph matching, which uses graduated assignment. They applied "softassign", a novel constraint satisfaction technique, to a new graph matching energy function that uses a robust, sparse distance measure between the links of the two graphs. Wu and Leon [64] proposed a two-pass greedy bipartite matching algorithm to determine the approximate solution of the stereo correspondence problem. Griffin [34] presented a bipartite graph matching method to determine the correspondences between 2-D projections of a 3-D scene. Roy and Cox [58] described a algorithm for solving the N-camera stereo correspondence problem.

2.4 Our Approach

The work in this paper is based on an optimal graph theoretic approach using a residual error function defined on the image plane to construct a bipartite graph, a corresponding flow network, and finally a maximum network flow that determines the correspondences between two images [13]. Unlike many other matching techniques, our method of network flow ensures that a maximal matching can be found. From the point of view of implementation, this matching technique can be implemented efficiently and in parallel, and has been successfully applied by others to matching problems involving graphs of large size (about 100,000 vertices) [41].

What is needed for correspondence matching among disparate, monocular images is a description of the affinity (or 2D/3D spatial relationship) between image features. The term "affinity" is first introduced by Ullman in [62] as a measure of the pairing likelihood of two objects in two sets. Here, we define the affinity as a measure of the degree to which candidates from different images consistently represent the projection of the same 3D point or the same 3D line.

Traditionally, two separate processing phases are employed to reconstruct 3D scene structure: feature matching and 3D triangulation. With this division of processing, it is very difficult to employ 3D information to measure the affinity between image features during the matching phase. We argue that it is better to combine matching and triangulation in an integrated manner. This is accomplished by introducing an affinity measure between image point and line features based on their distance from a hypothetical projected 3D pseudo-intersection point or line, respectively [14]. The ideas developed in this paper are an extension of our previous work [13, 14].

The challenge in combining matching and triangulation for image line features is that it is more difficult to describe the affinity between image lines than it is for image points. Line segment endpoints are not meaningful since there may exist significant fragmentation and occlusion in image line data, and therefore only the position and orientation of the *infinite image line* passing through a given line segment can be considered reliable. Moreover, this implies that at least three images are necessary to describe line affinity, since the projection planes for any pair of image lines in two images always intersect in a 3D line (Note: if parallel, the planes are said to intersect at infinity). Thus, no conclusive evidence about possible correspondences between infinite image lines may be derived from only two images.

One of the contributions of our work is an affinity measure for lines based on the rapid computation of a 3D pseudo-intersection line from a set of possibly corresponding image lines during the matching process. This approach leads to general maximum-weight bipartite matching techniques to deal with the 3D reconstruction problem without apriori specification of known image feature correspondences.

3 Measuring 2D Image Point amd Line Affinity

3.1 Measuring 2D Image Point Affinity from Two Images Via a 3D Pseudo-Intersection Point

Given a point $\boldsymbol{p}_1$ in image I_1, we seek its match $\boldsymbol{p}_2$ in another image I_2. Point $\boldsymbol{p}_2$ necessarily belongs on an *epipolar line* of image I_2 determined completely by $\boldsymbol{p}_1$, and vice versa. Most of the existing matching algorithms e.g. [5] directly utilize this 2D epipolar line constraint to determine the image point correspondences from two images. However, it is very difficult to employ 3D information to measure the affinity between image features from this 2D epipolar line constraint. We argue that it is better to combine matching and triangulation in an integrated manner.

The key observation is that for any pair of image points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ from two images I_1 and I_2, there exists a *$3D$ pseudo-intersection point*, defined as the point with the smallest sum of squared distances from it to the two projection lines of $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$. The physical meaning of the pseudo-intersection point is that ideally, if $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are corresponding image points from I_1 and I_2, then their pseudo-intersection point is a real $3D$ point recovered by the traditional triangulation constraint.

Given any pair of image points chosen at random, one from each image, the pair may or may not truly correspond to a single 3D point in the scene. The two image points always yield a 3D pseudo-intersection point in either case, but when this 3D point is projected back into each image it will only coincide reasonably well with the original pair of 2D image points if the points are a true correspondence, and will yield a very poor fit otherwise. Therefore, the distance between the projected pseudo-intersection point and the original pair of image points yields an affinity measure that signifies whether that pair of points forms a compatible correspondence.

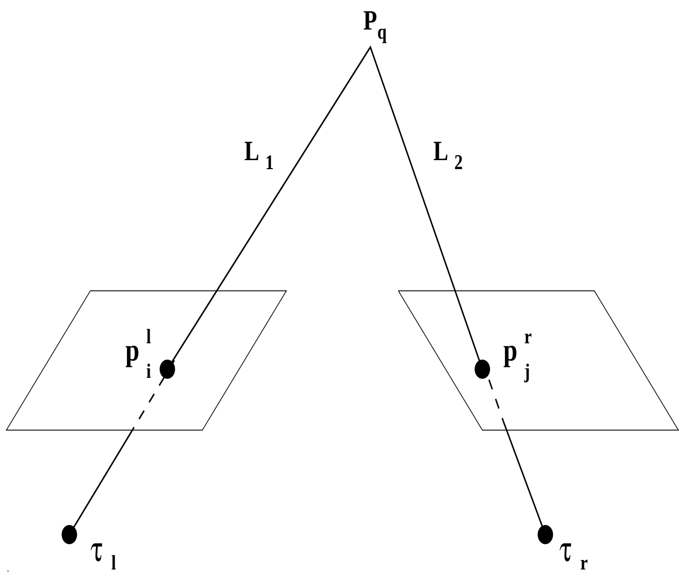


Figure 1: A triangulation process for a pair of images.

Let us now provide a formal specification of the affinity measure, with the reader referring to Figure 1. Given two poses $(\boldsymbol{R}_l, \boldsymbol{\tau}_l)$ and $(\boldsymbol{R}_r, \boldsymbol{\tau}_r)$ from two images I_l and I_r, any pair of $2D$ points $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$ $(i = 1, 2, \ldots, n_l;$ $j = 1, 2, \ldots, n_r)$ from I_l and I_r, define two $3D$ lines $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ such that $\boldsymbol{L}_1$ passes through points $\boldsymbol{p}_i^l$ and $\boldsymbol{\tau}_l$, and $\boldsymbol{L}_2$ passes through points $\boldsymbol{p}_j^r$ and $\boldsymbol{\tau}_r$. $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ are the *projection lines* of points $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$, respectively.

Suppose each projection line $\boldsymbol{L}_k$ $(k = 1, 2)$ is written as

$$\frac{x - x_k}{u_{xk}} = \frac{y - y_k}{u_{yk}} = \frac{z - z_k}{u_{zk}} \tag{1}$$

with unit direction vector $\boldsymbol{u}_k = (u_{xk}, u_{yk}, u_{zk})^T$.

Consider first how to compute an optimal $3D$ *pseudo-intersection point* $\boldsymbol{P}_q(x_q, y_q, z_q)$ with the smallest sum of distances from $\boldsymbol{P}_q$ to the two lines $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$. The error function can be defined [33] as

$$\begin{aligned} E = \quad & [(x_q - x_k)u_{yk} - (y_q - y_k)u_{xk}]^2 \\ & + [(x_q - x_k)u_{zk} - (z_q - z_k)u_{xk}]^2 \\ & + [(y_q - y_k)u_{zk} - (z_q - z_k)u_{yk}]^2 \end{aligned} \tag{2}$$

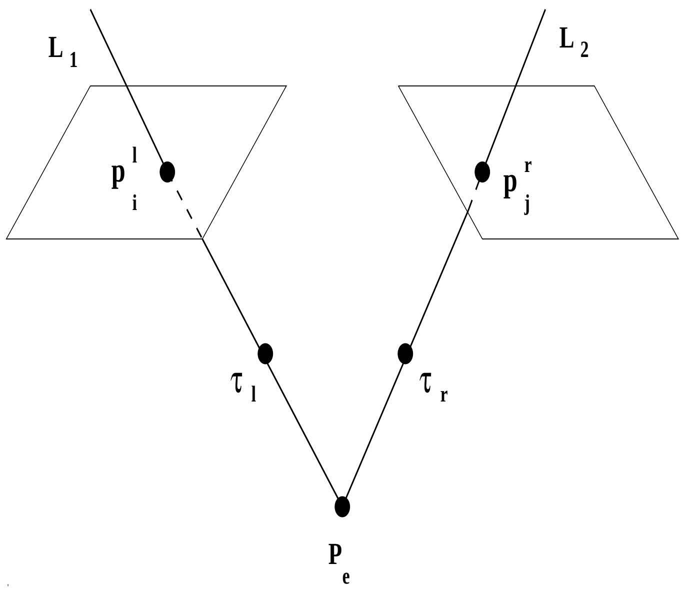


Figure 2: A wrong "negative" 3D point corresponding to a pair of image points that intersect behind one or both cameras. This is a detectable condition, even though it satisfies the definition of pseudo-intersection.

After setting $\frac{\partial E}{\partial x_q} = \frac{\partial E}{\partial y_q} = \frac{\partial E}{\partial z_q} = 0$, we obtain the optimal $3D$ pseudo-intersection point $\boldsymbol{P}_q$ [33]

$$\begin{bmatrix} x_q \\ y_q \\ z_q \end{bmatrix} = \left[\sum_{k=1}^{2} \boldsymbol{A}_k\right]^{-1} \left[\sum_{k=1}^{2} \left(\boldsymbol{A}_k \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} \right)\right] \tag{3}$$

where

$$\boldsymbol{A}_k = \begin{bmatrix} u_{yk}^2 + u_{zk}^2 & -u_{xk}u_{yk} & -u_{xk}u_{zk} \\ -u_{xk}u_{yk} & u_{xk}^2 + u_{zk}^2 & -u_{yk}u_{zk} \\ -u_{xk}u_{zk} & -u_{yk}u_{zk} & u_{xk}^2 + u_{yk}^2 \end{bmatrix}$$

As mentioned before, if $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$ are true corresponding image points, then $\boldsymbol{P}_q$ is the real $3D$ point to be recovered. However, there are four cases that are exceptions: (1) no $3D$ point could be obtained for $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$, because the two $3D$ lines $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ are parallel; (2) an incorrect "negative" $3D$ point could be obtained for $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$, due to the two $3D$ lines $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ intersecting behind one or both cameras, as shown in Figure 2; (3) a wrong "epipolar" $3D$ point $\boldsymbol{P}_w$ is obtained due to ambiguous correspondences, e.g. a typical problem with $\boldsymbol{p}_i^l$ corresponding to either $\boldsymbol{p}_j^r$ or $\boldsymbol{p}_w^r$ is shown in

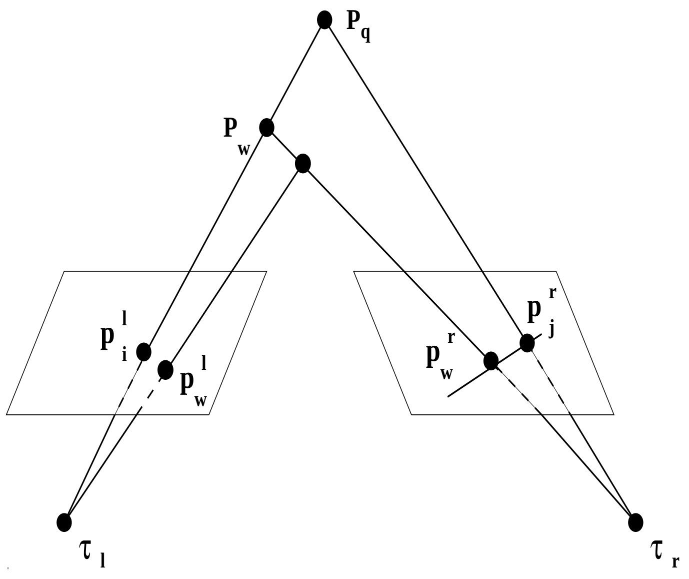


Figure 3: An error in correspondence produces a wrong 3D Pseudo-intersection point $\boldsymbol{P}_w$, or ambiguity in pairs of correspondences. $\boldsymbol{P}_q$ is the correct pseudo-intersectionpoint, but an incorrect correspondence using $\boldsymbol{P}^r_w$ along the correct epipolar line produces $\boldsymbol{P}_w$ as an incorrect psuedo-intersection point. If $\boldsymbol{P}^l_w$ is an existing candidate correspondence, then two pairs of ambiguous correspondences can be successfully resolved.

Figure 3. This case shows that a point $\boldsymbol{p}_i^l$ in image I_l could intersect with the projection line of more than one image point in image I_r; (4) a pseudo-intersection 3D point $\boldsymbol{P}_q$ is computed although $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$ don't correspond at all.

The first case, with parallel projection lines, is exceedingly rare, but is easily detected by examining whether a solution exists for Eq. 3. It also can be detected by examining whether the directions of the projection lines $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ are the same.

For the second case, as all pairs of image points from two images are considered initially as possible correspondences, some of those will intersect in their negative directions and satisfy the minimal distance condition to lines $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$, but are incorrect. Fortunately, it is easy to detect this kind of "negative" $3D$ point by examining the directions of rays from $\boldsymbol{\tau}_l$ to $\boldsymbol{p}_i^l$ and from $\boldsymbol{\tau}_l$ to $\boldsymbol{P}_q$ or rays from $\boldsymbol{\tau}_r$ to $\boldsymbol{p}_j^r$ and from $\boldsymbol{\tau}_r$ to $\boldsymbol{P}_q$ to make sure that they are the same.

The third case is the interesting one, caused by an incorrect correspondence, due to ambiguity. For example, as shown in Figure 3, suppose $\boldsymbol{p}_i^l$ corresponds to $\boldsymbol{p}_j^r$ with $\boldsymbol{P}_q$ as the correct $3D$ point. However, the known poses specify epipolar lines, and since $\boldsymbol{p}_w^r$ lies on the known epipolar line of $\boldsymbol{p}_i^l$ in image I_r, then both are plausible but ambiguous candidates for a correspondence match. Thus $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_w^r$ would intersect at a $3D$ point $\boldsymbol{P}_w$. However, this kind of ambiguity might be detected because $\boldsymbol{p}_w^l$ might correspond to another existing point $\boldsymbol{P}_w^r$ appearing in image I_r. For this case, the true (maximum) correspondences could be detected for the two sets of points, i.e., $\boldsymbol{p}_i^l$ corresponds to $\boldsymbol{p}_j^r$ and $\boldsymbol{p}_w^l$ corresponds to $\boldsymbol{p}_w^r$. Unfortunately, if the point $\boldsymbol{p}_w^l$ doesn't appear in the first image, it is difficult to resolve this inherent ambiguity. In such situations, a third image would greatly reduce such ambiguities.

For the fourth case, since it is an incorrect correspondence, the pseudo-intersection point $\boldsymbol{P}_q$ is located far from the two projection lines. This case is easily detected by its 2D affinity function defined below.

For any pair of image points ($\boldsymbol{p}_i^l$,$\boldsymbol{p}_j^r$), we project the "pseudo-intersection" point $\boldsymbol{P}_q$ into the two images I_l and I_r, to get the two projected image points $\boldsymbol{p}_i^{l'}(u_i', v_i')$ and $\boldsymbol{p}_j^{r'}(u_j', v_j')$. Finally, we compute the error functions E_{ij}^l and E_{ij}^r:

$$E_{ij}^l = \|\boldsymbol{p}_i^l - \boldsymbol{p}_i^{l'}\|_2, E_{ij}^r = \|\boldsymbol{p}_j^r - \boldsymbol{p}_j^{r'}\|_2 \tag{4}$$

and define a 2D *point affinity function* $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$ is defined as

$$sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r) = e^{-(E_{ij}^l + E_{ij}^r)/2} \tag{5}$$

The criterion underlying $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$ is that the best estimate for any 3D pseudo-intersection point is the point that minimizes the sum of the squared distances between the predicted image location of the computed 3D point and its actual image locations in the first and second images. If $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$ = 0, it means that $\boldsymbol{p}_i^l$ is not a possible match for $\boldsymbol{p}_j^r$; if $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$ = 1, it means that $\boldsymbol{p}_i^l$ is a perfect match for $\boldsymbol{p}_j^r$.

3.2 Measuring 2D Line Affinity from Three Images via a 3D Pseudo-Intersection Line

Here, we develop an analagous line pseudo-intersection measure, similar to the 2D affinity measure between image point features in the last subsection. Given the poses of three images, a 2D affinity measure among image line segments is developed for the problem of determining image line correspondences, while simultaneously computing the corresponding $3D$ lines. The challenge in combining matching and triangulation for image line features is that it is more difficult to describe the affinity between image line segments. Since it is well-known that line segment endpoints are prone to error due to fragmentation and unreliable terminations in image line data, only the position and orientation of the infinite image line passing through a given line segment of sufficient length can be considered reliable. Moreover, this implies that at least three images are necessary to describe affinity, since the projection planes for any pair of image lines in two images always intersect in a 3D line (if parallel, the planes are said to intersect at infinity), and thus no conclusive evidence about possible correspondences between infinite image lines may be derived from only two images.

Given three images and their corresponding camera poses, we assume that three line segments are chosen at random, one from each image, so that the set may or may not truly correspond to a single 3D line in the scene. For any triplet of image lines from three images, there exists a 3D *pseudo-intersection line* L with the smallest sum of squares of the mutual moments of L with respect to the two projection lines of the two endpoints for each image line. As this computation will be performed many times on images with large numbers of line segments as we search for correct correspondences, we need it to be computationally efficient, even if this speed is achieved at the expense of accuracy. Here, the linear line reconstruction algorithm presented in [12] is employed to achieve a closed-form solution for the best

3D pseudo-intersection line. It has been shown in [12] that the algorithm is fast and efficient.

For any triplet of image line segments from three images, we wish to compute an affinity value that measures the degree to which these lines are consistent with the hypothesis that they are all projections of the same linear 3D scene structure. To do this, we first use the line reconstruction algorithm presented in [12] to compute their pseudo-intersection line L, and then project $\boldsymbol{L}$ back into each image to get three infinite image lines $\boldsymbol{l}_i (i = 1, ..., 3)$.

Suppose $\boldsymbol{l}_i$ is represented by the equation

$$f_i u + g_i v + h_i = 0$$

in pixel coordinates (u, v), and that the endpoints of the original 2D line segment in image I_i are (u_a, v_a) and (u_b, v_b). A natural measure of the distance from the line segment to the projected pseudo-intersection line l_i is the sum of absolute pixel distances from the line segment endpoints to l_i, that is

$$r_i = \frac{\mid f_i u_a + g_i v_a + h_i \mid + \mid f_j u_b + g_i v_b + h_i \mid}{\sqrt{f_i^2 + g_i^2}} \quad . \tag{6}$$

If the three image line segments actually are a true correspondence of a single linear 3D structure, we can expect all of them to lie “close” to their respective reprojections of the pseudo-intersection line, where closeness is judged based on our knowledge of the error characteristics of the line segment extraction process and the level of noise in the image. On the other hand, if the image line segments do not correspond to a linear scene structure, their distance from the projected pseudo-intersection line will be large, which is true of most of the line triplets (barring accidental alignments). The distance is greater to the extent that the chosen lines are truly geometrically incompatible.

Based on the above distance measure, the *2D line affinity* value $sfl(\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3)$ for a triplet of image line segments from three images is defined as

$$sfl(\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3) = e^{-(\sum_{i=1}^{3} r_i)/6} \tag{7}$$

where $\sum_{i=1}^{3} r_i/6$ can be interpreted as the average distance from the set of image line segment endpoints to their respective projected pseudo-intersection lines. If $sfl(\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3) = 0$, it means that $\boldsymbol{l}_1$,$\boldsymbol{l}_2$, and $\boldsymbol{l}_3$ are not compatible at all; if $sfl(\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3) = 1$, it means tha $\boldsymbol{l}_1$,$\boldsymbol{l}_2$, and $\boldsymbol{l}_3$ are perfectly compatible.

4 3D Reconstruction Algorithms without Correspondences Based on the Weighted Bipartite Matching Technique

Traditional correspondence matching techniques use only 2D pixel-level information. These 2D image analysis techniques encounter significant difficulty in recovering correct correspondences of image features, since they do not consider the important 3D information captured via our pseudo-intersection points or lines.

In the previous section, we developed two affinity measures between image features, $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$ in Eq. 5 for image points and $sfl(\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3)$ in Eq. 7 for image lines. The two affinity measure functions contain significant information about potential correspondences between image features. The important question that immediately follows is how to use this information in a reliable process to determine image feature correspondences. Weighted bipartite matching [41, 42] is a mature mathematical framework for solving matching problems using our affinity measure.

In this section, we will show how the problem of image feature matching can be formulated as a maximum-weight bipartite matching problem by using the 2D image point affinity function in Eq. 5 and the 2D image line affinity function in Eq. 7. An efficient graph-based algorithm for matching and reconstruction is developed to determine image feature correspondences while simultaneously recovering 3D features.

4.1 Formulation as a Maximum-Weight Bipartite Matching Problem

Given the two sets of image points $\mathrm{L} = \{\boldsymbol{p}_i^l \mid i = 1, 2, ..., n_l\}$ from image I_l and $R = \{\boldsymbol{p}_j^r \mid j = 1, 2, ..., n_r\}$ from image I_r, an undirected weighted graph $G = (V, E)$ can be constructed as follows: $V = L \cup R, E = \{e_{ij}\}$. Each edge e_{ij} $(i = 1, 2, ..., n_l; j = 1, 2, ..., n_r)$ corresponds to a weighted link between $\boldsymbol{p}_i^l$ in I_l and $\boldsymbol{p}_j^r$ in I_r, whose weight $w(e_{ij})$ is equal to the affinity between $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$, i.e. $w(e_{ij}) = sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$. Obviously, the graph arising in such a case is a weighted bipartite graph by construction, since two points in the same image cannot be linked.

Given a set of line segments $\boldsymbol{l}_\alpha$,$\boldsymbol{l}_\beta$, and $\boldsymbol{l}_\gamma$ in a triplet of images I_1, I_2 and I_3, two undirected bipartite graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ can be constructed as follows. First, generate two vertex sets V_1 and V_2 such that $V_1 = I_1 \cup I_2$ and $V_2 = I_2 \cup I_3$. Next, for all feasible matches among any three image lines $\boldsymbol{l}_\alpha$, $\boldsymbol{l}_\beta$ and $\boldsymbol{l}_\gamma$, one from each image, generate their edges $e^1_{\alpha\beta} \in E_1$ and $e^2_{\beta\gamma} \in E_2$ with weights equal to the affinity measure $sfl(\boldsymbol{l}_\alpha, \boldsymbol{l}_\beta, \boldsymbol{l}_\gamma)$, as defined in the last section, i.e. $w(e^1_{\alpha\beta}) = w(e^2_{\beta\gamma}) = sfl(\boldsymbol{l}_\alpha, \boldsymbol{l}_\beta, \boldsymbol{l}_\gamma)$. Note that in general this could involve taking all triplets of image line segments, one from each image, unless domain specific information is used to prune the set of possible matches down to a smaller feasible set. Often such information should be available through domain constraints.

It should be noted that due to the fragmentation of image lines, multiple competing edges could exist between the same two nodes in either graph. For example, suppose there exists a possible correspondence among the line segments $\boldsymbol{l}_\alpha$, $\boldsymbol{l}_\beta$, and $\boldsymbol{l}_\gamma$ from the three images respectively, and another possible correspondence between $\boldsymbol{l}_\alpha$, $\boldsymbol{l}'_\beta$, and $\boldsymbol{l}_\gamma$. It would seem then, that two edges between $\boldsymbol{l}_\alpha$ and $\boldsymbol{l}_\beta$ are needed, one to store the weight for $sfl(\boldsymbol{l}_\alpha, \boldsymbol{l}_\beta, \boldsymbol{l}_\gamma)$ and one for $sfl(\boldsymbol{l}_\alpha, \boldsymbol{l}_\beta, \boldsymbol{l}'_\gamma)$. In practice, we remove these trivial conflicts at graph creation time by checking if an edge already exists between two nodes before adding a new one. If the affinity value of the new edge is larger than the edge already there, then the old edge is replaced by the new one, otherwise it is left alone.

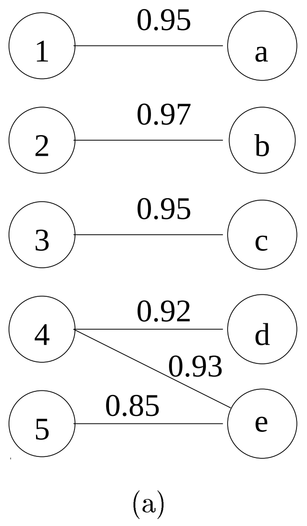
From the previous subsections, we know that for any pair of image points $\boldsymbol{p}^l_i$ and $\boldsymbol{p}^r_j$, there is a weighted link e_{ij} between $\boldsymbol{p}^l_i$ and $\boldsymbol{p}^r_j$ in the weighted bipartite graph G. Similarly, for any triplet of image line segments $\boldsymbol{l}_\alpha$, $\boldsymbol{l}_\beta$, and $\boldsymbol{l}_\gamma$, there is a weighted link $e^1_{\alpha\beta}$ between $\boldsymbol{l}_\alpha$ and $\boldsymbol{l}_\beta$ in the first bipartite graph G_1 and a weighted link $e^2_{\beta\gamma}$ between $\boldsymbol{l}_\beta$ and $\boldsymbol{l}_\gamma$ in the second bipartite graph G_2. Ideally, if the image features (points/lines) are in true correspondence then their weights $w(e_{ij})$, or $w(e^1_{\alpha\beta})$ and $w(e^2_{\beta\gamma})$ should be equal to the maximum weight of 1; thus they significantly contribute to the final matching to be determined, and the number of total image feature (point/line) correspondences is equal to the size of the matching. Due to the errors in some of the camera poses and the locations of the extracted image points or line segments, however, the weights $w(e_{ij})$, or $w(e_{\alpha\beta})$ and $w(e_{\beta\gamma})$ will be below 1, but often can be expected to be high (i.e. approach 1).

On the other hand, from graph theory, we know that given an undirected graph, a *matching* is a subset of edges $M \subseteq E$ such that for all vertices $v \in V$, at most one edge of M is incident on v. A vertex $v \in V$ is matched by M if some edge in M is incident on v; otherwise, v is unmatched. The *maximum-weight matching* is a matching M_w of size $\mid M_w \mid$ such that the sum of the weights of the edges in M_w is maximum over all possible matchings. Therefore, the image feature correspondences to be determined correspond to the maximum-weight matching in the bipartite graphs G for determination of image point correspondences, or G_1 and G_2 for determination of image line correspondences.

4.2 Reduction to the Maximum-Flow Problem

As discussed from Subsection 4.1, the correspondence problem of image points and lines can be considered as the problem of finding the maximum-weight matching in the weighted bipartite graphs. The remaining question is how to find the maximum-weight matching in the weighted bipartite graphs.

If each edge has a unit weight in the bipartite graph, then we get the unweighted bipartite matching problem, which is to find a matching of maximum cardinality. The above image feature matching problem for image points and lines could be reduced to the unweighted matching problem by setting all the weights in the bipartite graph to be 1 if $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r) \geq T_p$ for image points $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$, or if $sfl(\boldsymbol{l}_\alpha, \boldsymbol{l}_\beta, \boldsymbol{l}_\gamma) \geq T_l$ for image line segments $\boldsymbol{l}_\alpha, \boldsymbol{l}_\beta$, and $\boldsymbol{l}_\gamma$. Here, the thresholds T_p and T_l would be chosen empirically. For the weighted bipartite graph shown in Figure 4(a), its unweighted counterpart is shown in Figure 4(a) by setting all the weights in the bipartite graph to be 1 if $sfp(\boldsymbol{p}_i^1, \boldsymbol{p}_j^2) \geq T_p = 0.8$.



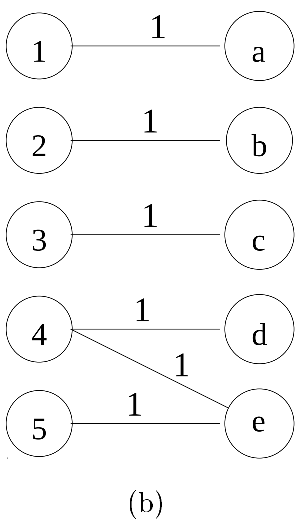


Figure 4: The weighted and unweighted bipartite graphs: (a) weighted bipartite graph; (b) unweighted bipartite graph.

The problem of image feature matching seems on the surface to have little to do with flow networks, but it can in fact be reduced to a maximum-flow problem. By relating the unweighted matching problem for bipartite graphs to the max-flow problem for simple networks, the matching problem becomes simpler, and the fastest maximum flow algorithm can be used to find the maximum matching, which was discussed in [13]. In order to reduce the problem of a maximum matching in the bipartite graph G to a maximum flow problem in the flow network G', the trick is to construct a flow network in which flows represent correspondences. We build a corresponding flow network $G' = (V', E')$ for the bipartite graph G as follows: Let the source s and sink t be new vertices not in V, let $V' = V \cup \{s, t\}$, and let the directed edges of G' be given by

$$\begin{aligned} E' = \ & \{(s, v_i') : v_i' \in L\} \cup \{(v_i', v_j') : v_i' \in L, v_j' \in R, \\ & (v_i', v_j') \in E\} \cup \{(v_j', t) : v \in R\} \end{aligned}$$

and finally, assign unit flow capacity to each edge in E.

Further, it has been shown that a maximum matching M_w in a bipartite graph G corresponds to a maximum flow in its corresponding flow network G'. Therefore, the unweighted image feature correspondence problem is exactly equivalent to finding the maximum flow in $G' = (V', E')$, and we can compute a maximum matching in G by finding a maximum flow in G'. The main advantage of formulating the image feature correspondence problem as the unweighted bipartite matching problem is that there exist very fast algorithms (e.g. Goldberg's algorithm is $O(|V||E|log|V|)$), which can be implemented in an efficient and parallel way to find the maximum matching in the unweighted bipartite graph.

4.3 Solving for the Maximum-Weight Bipartite Match

The main disadvantage of the unweighted bipartite matching formulation is that it is crucial to choose an appropriate value for the threshold T_p for the image point correspondence problem and T_l for the image line segment correspondence problem *before the unweighted bipartite matching algorithm is performed*. If T_p or T_l is too small, more outliers will be created; if T_p or T_l is too large, it will filter out too many correct correspondences. For example, as shown in Figure 4(a), if we choose $T_p = 0.9$, then the correct correspondence $(5, e)$ could be filtered. In this case, we would miss the matching $(5, e)$ and could not then disambiguate the matchings $(4, d)$ and $(4, e)$ for the left image point "4". Therefore, it is necessary to deal with the general maximum-weight bipartite matching problem, which is the generalization of the unweighted bipartite matching problem. Although the weighted matching problem is not characterized by maximum flows in terms

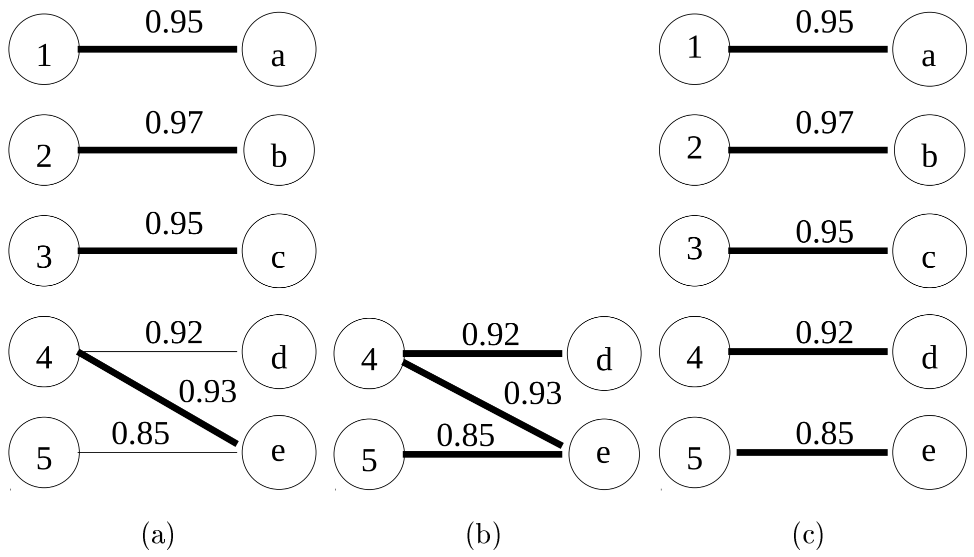


Figure 5: The symmetric difference operator of M and P: (a) matching M; (b) augmenting path P wrt. M; (c) $M \oplus P$.

of augmenting paths, it indeed can be solved based on exactly the same idea: start with any empty matching, and repeatedly discover augmenting paths. In the following, we focus on how to find the maximum-weight matching in the weighted bipartite graph.

Consider the matching M shown in Figure 5(a). The edges $(1, a)$, $(2, b)$, $(3, c)$, and $(4, e)$ are matched, and the edges $(4, d)$ and $(5, e)$ are unmatched. Given a matching M in a bipartite graph $G = (V, E)$, a simple path in G is called an *augmenting path with respect to matching M* if its two endpoints are both unmatched, and its edges alternate between $E - M$ and in M. The augmenting path $P = \{(5, e), (e, 4), (4, d)\}$ with respect to matching M is shown in Figure 5(b). Endpoints 5 and d are unmatched, and the path consisting of alternating edges (5,e) in $E - M$, (4,e) in M, and finally (4,d) in $E - M$.

Let p denote an augmenting path with respect to matching M, and P denote the set of edges in p, then $M \oplus P$ is called the *symmetric difference* of M and P. $M \oplus P$ is the set of elements that are in one of M or P, but not both, i.e. $M \oplus P = (M - P) \cup (P - M)$. It can be shown that $M \oplus P$ has the following properties: (1) it is a matching; (2) $\mid M \oplus P \mid = \mid M \mid +1$. The symmetric difference $M \oplus P$ is shown in Figure 5(c), i.e. $M \oplus P = \{(1, a), (2, b), (3, c), (4, e), (5, d)\}$, and $\mid M \oplus P \mid$ = 4+1=5.

For the matching M, its *total weight of matching M* is defined as

$$w(M) = \sum_{e \in M} w(e)$$

Let M' be a set of edges; then an *incremental weight* $\Delta M'$ is defined as the total weight of the unmatched edges in M' minus the total weight of the matched edges in M':

$$\Delta M' = w(M' - M) - w(M' \cap M)$$

From the definition of incremental weight, we know that for an augmenting path p with respect to M, then ΔP gives the net change in the weight of the matching after augmenting p:

$$w(M \oplus P) = w(M) + \Delta P$$

Intuitively, we can use an iterative algorithm to construct a maximum-weight matching. Initially, the matching M is empty. At each iteration, the matching M is increased by finding an augmenting path of maximum incremental weight. This is repeated until no augmenting path with respect to matching M can be found. It has been proven that repeatedly performing augmentations using augmenting paths of maximum incremental weight, yields a maximum-weight matching M_w [41].

The remaining problem is how to search for augmenting paths with respect to matching M in a systematic and efficient way. Naturally, a search for augmenting paths must start by constructing alternating paths from the unmatched points. Because an augmenting path must have one unmatched endpoint in L and the other in R, without loss of generality, we can start growing alternating paths only from unmatched vertices of L. We may search for all possible alternating paths from unmatched vertices of L simultaneously in a breadth-first manner. Here, an efficient Gabor's N-cubed weighted matching algorithm [41] is used to compute the maximum-weight matching in the weighted bipartite graph. This algorithm has two basic steps: (1) to find a shortest path augmentation from a subset of left vertices in L to a subset of right vertices in R; (2) to perform the shortest augmentation. The algorithm is very efficient; more implementation are discussed in [41].

Since the number of matched vertices increases by two each time, this takes at most $\frac{n}{2}$ augmentations. It has been shown that for a matching M of size k of maximum weight among all matchings of size at most k, if there exists a matching M^* of maximum weight among all matchings in G, and $w(M^*) \geq w(M)$, then M has an augmenting path of positive incremental weight. Therefore, the image feature correspondence problem can be exactly reduced to finding the maximum-weight matching in the weighted bipartite graph.

In summary, the general matching and reconstruction algorithm for image point correspondences can be achieved by the following steps:

Step 1: compute a pseudo-intersection point for each pair of image points $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$.

Step 2: calculate the value of $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$ in Eq. 5 for each pair of image points $\boldsymbol{p}_i^l$ and $\boldsymbol{p}_j^r$.

Step 3: remove the pair for further graph-based matching analysis if its $sfp(\boldsymbol{p}_i^l, \boldsymbol{p}_j^r)$ value is less than certain predefined threshold.

Step 4: determine if the pair should not be added into a weighted bipartite graph in Step 5 with respect to incorrect Cases 1 and 2, which were discussed in Section 3.1.

Step 5: construct a weighted bipartite graph $G = (V, E)$ for image points.

Step 6: find the maximum weighted matching M_w for G.

Step 7: determine image point correspondences and their corresponding 3D points from the maximum matching M_w.

Similarly, the general matching and reconstruction algorithm for image line correspondences can be achieved by the following steps:

Step 1: compute a pseudo-intersection line for each triplet of image lines $\boldsymbol{l}_1, \boldsymbol{l}_2$, and $\boldsymbol{l}_3$.

Step 2: calculate the value of $sfl(\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3)$ in Eq. 7 for each triplet of image lines $\boldsymbol{l}_1, \boldsymbol{l}_2$, and $\boldsymbol{l}_3$.

Step 3: remove the triplet for further graph-based matching analysis if its $sfl(\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3)$ value is less than certain predefined threshold.

Step 4: construct two weighted bipartite graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ for image line segments.

Step 5: find the maximum weighted matching M_w for G_1 and G_2.

Step 6: determine image line correspondences and their corresponding 3D lines from the maximum matching M_w.

It should be noted that Step 3 in the above two matching algorithms is not necessary, but can filter out a great number of incorrect correspondences that don't correspond at all, thus improve computational efficiency and running time of the two matching algorithms since it can reduce the numbers of vertices and edges, i.e. the size of the bipartite graphs. Incorrect correspondences that are not filtered in Step 3, will be determined by our weighted bipartite matching process.

5 Experimental Results for Correspondence of Image Points

In this section, we present experiments to characterize the performance of our approach to 3D point reconstruction while simultaneously determining correspondences, based on the affinity function defined in Eq. 5. We will examine the performance in terms of the number of the recovered image point correspondences, and the distance between each triangulated 3D pseudo-intersection point and its actual 3D point. In all the experiments, we assume that both the intrinsic camera parameters and poses are known. The algorithm uses two sets of image points separately extracted from two images as input, and produces a set of image point correspondences and their corresponding triangulated 3D points.

5.1 Synthetic Data

The synthetic experiments are performed on a set of synthesized 3D points representing a rigid object. To evaluate performance with known ground truth, a set of 40 3D points were randomly generated from the object and projected into two images. The image point locations for each image were corrupted by Gaussian noise. Noise for each image point location was assumed to be zero-mean, identically distributed, and independent. The standard deviation ranges from 1.0 pixel to 5.0 pixels. In order to examine how the robustness of matching is affected by missing points (i.e. no correct correspondence in the other image), 16 sets were generated with different percentages of missing points ranging from $\frac{0}{40}\%$ to $\frac{15}{40}\%$. For each of these reduced point sets, 100 trials of noisy samples were used to spatially perturb the remaining points for each of the five levels of noise. For each sample of the same set, the number of missing points is the same, i.e. the same percentage of image points were randomly deleted. The algorithm was run on each of the samples, and the number of incorrect image point correspondences was computed for each sample run. Figure 6 shows the average number of incorrect correspondences for each noise level.

As shown in Figure 6, the algorithm works very well if the number of missing points is 0, i.e. each 3D point to be recovered is visible in both images. On average, there is only one incorrect correspondence even for the highest noise level of 5 pixels. For lower levels of noise ranging from 1

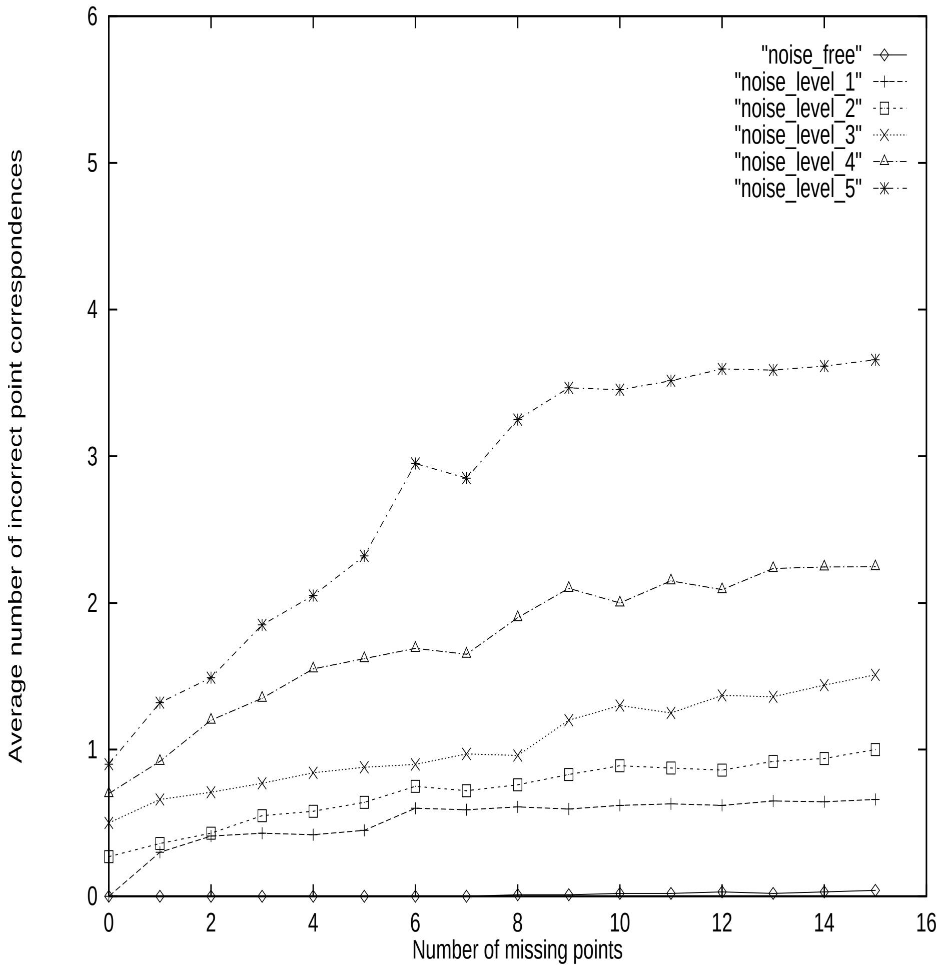


Figure 6: Performance of the image point matching algorithm against five levels of noise with different numbers of missing points. The average number of incorrect correspondences is shown for the true set of 3D points. Noise level k means all points were perturbed with Gaussian noise of standard deviation k.

pixel to 3 pixels, there is litte effect on the performance of the algorithm for different numbers of missing points. For the higher levels of noise, the number of incorrect correspondences increases linearly as the difference in the sizes of image points from two images increases. From Figure 6, we can see that on average, the number of incorrect correspondences rises about 3% at any of the noise levels. Therefore, our experiments have shown that the algorithm can tolerate a significant difference in the number of image points from two images and is robust against a reasonable level of noise.

5.2 PUMA Sequence

The 3D point reconstruction algorithm with unknown correspondences is applied to the set of real images referred to as the PUMA sequence [49], since the images were acquired by a camera mouted on a PUMA robot arm. The image sequences were captured with a SONY B/W AVC D-1 camera with an effective field of view of $41.7°$ (fovx: field of view x-axis) by $39.5°$ (fovy: field of view y-axis) and the image resolution is 256×242. Thirty frames were taken over a total angular displacement of 116 degrees. The maximum displacement of the camera in these twenty frames is approximately 2 feet along the world y-axis and 1 foot along the world x-axis.

For each image, line segments were first extracted by the Boldt algorithm [9], and then 2D corner image points were computed by calculating the intersection point between any pair of nearby image line segments. Figure 7 shows two sets of extracted and unmatched image points from the $1st$ frame and the $10th$ frame, respectively. There is a difference in the number of image points from the two images, since the $1st$ frame has 113 image points while the $10th$ frame has 107 image points. There are two kinds of error sources: the 2D image point locations and the estimated camera parameters. The noise in the image points is due to many typical factors such as camera distortion and errors in the image line extraction algorithm. The noise in the camera parameters is mainly due to errors in camera calibration. As seen in Figure 7, some image points were extracted in the $1st$ frame, but not in the $10th$ frame, and vice versa. Thus, it is a general 3D point reconstruction problem with unknown image point correspondences.

Figure 8 shows a subset of 43 correct image point correspondences determined from the two frames. The corresponding 3D points reconstructed by the algorithm are reported in Table 1. This experiment uses the ground truth 3D data supplied in Kumar's thesis [49]. Note that here we only reported the comparisons between the reconstructed 3D points and their ground truth data for those 3D points whose ground truth coordinates are available. As shown in Table 1, there is only one incorrect correspondence labeled 21, where the correspondence is of two different image points from the 1st frame and $10th$ frame. Point 21 in Frame 1 is on the lower of two rectangles (the upper left corner), while Point 21 in Frame 10 is on the upper rectangle (the lower left corner). Although they are very close in the images, the absolute error between the triangulated 3D point recovered by

Table 1: 3D point reconstruction error for the PUMA ROOM data (average 3D error: 0.43 feet without incorrect point 21.

Line	Actual 3D point			Computed 3D points			Error
	x	y	z	x	y	z	(feet)
1	-6.06	-0.47	13.70	-5.94	-0.52	13.48	0.26
2	-6.06	-0.47	11.56	-5.89	-0.55	11.18	0.43
3	-6.06	-0.47	16.81	-6.87	-0.50	16.62	0.83
4	-6.06	-0.47	14.68	-6.88	-0.49	14.50	0.84
5	-8.58	-0.47	16.81	-8.53	-0.48	16.64	0.18
6	-8.58	-0.47	14.68	-8.54	-0.45	14.59	0.11
7	0.00	4.05	20.82	0.20	4.04	18.89	1.94
8	0.00	8.13	13.51	0.13	8.21	13.32	0.25
9	0.00	6.81	14.91	0.17	6.89	14.64	0.33
10	0.00	7.31	13.51	0.15	7.38	13.28	0.28
11	0.00	7.00	13.89	0.19	7.09	13.60	0.36
12	0.00	7.00	11.76	0.18	7.11	11.47	0.36
13	0.00	5.36	13.89	0.11	5.39	13.76	0.17
14	0.00	4.69	14.89	0.18	4.72	14.66	0.30
15	0.00	4.66	16.00	0.27	4.72	16.19	0.33
16	0.00	4.98	11.95	0.25	5.07	11.58	0.46
17	0.00	4.92	14.11	0.23	4.98	13.77	0.41
18	0.00	4.25	11.96	0.34	4.22	11.42	0.64
19	-3.32	9.01	7.03	-3.17	9.23	6.35	0.73
20	-1.45	9.01	3.13	-1.08	9.18	2.27	0.95
21	-3.02	3.80	11.28	0.68	4.26	2.33	9.69
22	-6.32	8.11	0.00	-6.13	8.27	-0.85	0.88
23	-4.20	8.07	0.00	-4.00	8.19	-0.64	0.68
24	-6.35	6.49	0.00	-6.20	6.56	-0.55	0.58
25	-1.77	2.86	0.00	-1.52	2.87	-0.68	0.73
26	-7.26	8.09	0.00	-7.11	8.17	-0.47	0.50
27	-7.26	6.45	0.00	-7.15	6.50	-0.25	0.27
28	-4.82	-0.47	17.82	-4.68	-0.52	17.64	0.23
29	-4.81	-0.47	14.82	-4.67	-0.54	14.56	0.30
30	-4.81	-0.47	16.95	-4.66	-0.52	16.73	0.27
31	-4.43	-0.47	11.63	-4.22	-0.55	11.30	0.40
33	-6.44	-0.47	16.95	-6.35	-0.52	16.74	0.23
34	-6.94	-0.47	19.45	-6.86	-0.52	19.24	0.23
35	-6.94	-0.47	17.82	-6.88	-0.50	17.66	0.18
36	-7.53	-0.47	17.54	-7.48	-0.49	17.38	0.16
37	-3.32	9.01	15.03	-3.42	9.04	15.08	0.12
38	0.00	5.37	11.76	0.35	5.48	11.17	0.70
39	0.00	4.10	14.09	0.24	4.15	13.76	0.41
40	-1.43	3.64	0.00	-1.21	3.67	-0.55	0.59
41	-4.23	6.45	0.00	-4.05	6.53	-0.61	0.64
43	-6.87	0.11	13.75	-6.84	0.09	13.71	0.05

this incorrect correspondence and the original 3D point associated with image point 21 in the first frame has large error, 9.69 feet. For the other 42 correct correspondences, the average distance between the triangulated 3D points and ground truth data is 0.43 feet.

5.3 RADIUS Image Set

The 3D point reconstruction algorithm without apriori correspondences is also applied to the RADIUS image set. This experiment uses data supplied through the ARPA/ORD RADIUS project(Research and Development for Image Understanding Systems) [4]. The images and camera parameters used in this experiment were the "model board 1" data set distributed with version 1.0 of the RCDE (RADIUS Common Development Environment) software package [54]. The image size is approximate 1320×1035 pixels. Unlike the PUMA image sequence used in last subsection, each pair of images from this data set were taken from two disparate views. Eight images were provided in this data set.

Again, for each image, line segments and 2D corner points were extracted as part of an automated building detection algorithm [17]. These 2D corner points are thus extracted in a different manner than those in the PUMA sequence. Here the points to be matched are the corners of building polygons. Figures 9 and 10 show two sets of extracted and unmatched image points from images J3 and J7, respectively. It should be noted that J3 and J7 have two different numbers of missing points although they have exactly the same number of image points, i.e. 186 points. Again, both the 2D image points and the camera parameters are noisy. The noise in the image points is again due to errors in point localization and camera calibration.

Our algorithm recovered 61 correspondence rooftop polygon points, all of them correct (Figures 11 and 12). The corresponding 3D points reconstructed by the algorithm are reported in Table 2. This experiment uses the ground truth 3D data supplied in the "model board 1" data set Here we only reported the comparisons between the reconstructed 3D points and their ground truth data for those 3D points whose ground truth coordinates are available. From Table 2, we can see that for some image correspondences such as 20, 21, 34, 35, 36, 49, 50, and 57, the triangulated 3D points have large errors although their correspondences are determined correctly by our algorithm. This is due mainly to the errors in the locations of rooftop polygon points, since it is well known that these 2D errors have a significant effect on the triangulated 3D data, especially when there are only two images [17]. Some of the increased size of errors can be attributed to 2D corners being "moved" due to shadows in one of the views (e.g. point 49 which produced the largest error). In order to improve overall accuracy, more images are required. Nevertheless, the results are quite good, with the average distance between the triangulated 3D points and ground truth data across 61 correct image point correspondences being 0.45 feet.

6 Experimental Results for Image Lines

In this section, we will demonstrate the performance of the 3D line reconstruction algorithm with unknown correspondences. It should be noted that the accuracy of the triangulated 3D lines depends upon the performance of the line reconstruction algorithm employed. Here, we use a fast line reconstruction algorithm with computational efficiency and robustness against noise [12]. Therefore, in the following, we will simply report a comparison between the triangulated 3D lines and their ground-truth data, and concentrate instead on characterizing the performance of the algorithm for determining the 2D line correspondences that are used for 3D line reconstruction. Thus, more detailed experiments are reported in terms of the number of the recovered image line correspondences. In all the experiments presented here, we again assume that both the intrinsic camera parameters and poses are known. Unlike the previous section, image lines are extracted directly as image features, and the algorithm uses three sets of image lines across three images as input, computing the image line correspondences and their corresponding triangulated 3D lines as output.

6.1 Synthetic Data

Simulations were performed on a set of synthetic 3D lines representing a rigid body. A set of 40 3D lines were randomly generated from an object, and projected into three images. The two endpoints of each image line segment in three images were corrupted by Gaussian noise. Noise for each image line segment endpoint was assumed to be zero-mean, identically distributed, and independent. The standard deviation of line endpoint noise ranges from 1.0 pixel to 5.0 pixels. In order to examine how the number of incorrect correspondences is affected by the number of missing image line segments, 16 sets of 100 noisy line samples were created for each level of noise, in terms of 16 different percentages of missing lines ranging from $\frac{0}{40}\%$ to $\frac{15}{40}\%$. For each sample of the same set, the number of missing lines is the same, i.e. the same percentage of image lines were randomly deleted. The algorithm was run on each of the samples, and the average number of incorrect image line correspondences was computed across samples used. Figure 13 shows 16

Table 2: 3D point reconstruction error for the RADIUS image data.

Line	Actual 3D point			Computed 3D points			Error
	x	y	z	x	y	z	(feet)
1	15.79	22.79	-1.15	16.12	22.79	-0.87	0.44
2	15.77	16.40	-1.15	16.01	16.51	-1.33	0.32
3	20.51	16.39	-1.15	20.74	16.48	-1.39	0.34
4	20.52	22.78	-1.15	20.83	22.76	-0.93	0.38
5	9.90	39.36	-1.21	10.45	39.46	-1.32	0.56
6	9.90	40.13	-1.21	10.45	40.16	-1.29	0.55
7	5.89	40.13	-1.21	6.26	40.17	-1.00	0.43
8	5.89	39.35	-1.21	6.26	39.44	-1.01	0.43
15	17.55	7.14	-0.16	17.79	7.17	0.46	0.67
16	17.52	0.76	-0.16	17.77	0.88	0.46	0.68
17	20.08	0.75	-0.16	20.32	0.80	-0.35	0.31
18	20.11	7.13	-0.16	20.34	7.14	-0.18	0.23
20	17.26	10.35	-0.22	17.53	10.48	1.06	1.31
21	17.27	13.36	-0.22	17.56	13.56	1.05	1.32
22	20.29	13.35	-0.22	20.52	13.52	0.06	0.40
23	22.92	11.25	-0.97	22.98	11.49	-0.86	0.27
24	1.60	23.56	-0.43	1.86	23.44	-1.06	0.69
25	1.61	26.53	-0.43	1.85	26.61	-1.21	0.82
26	-4.41	26.55	-0.43	-4.04	26.69	-0.65	0.45
27	-4.42	23.58	-0.43	-4.04	23.47	-0.49	0.40
28	1.64	27.16	-0.78	1.86	27.21	-1.29	0.56
29	1.61	23.38	-0.78	1.86	23.40	-1.04	0.36
30	4.49	26.52	-0.55	4.75	26.55	-0.60	0.27
31	4.34	16.55	-0.55	4.62	16.57	-0.53	0.28
32	14.33	16.40	-0.55	14.60	16.45	-0.65	0.29
33	14.49	26.31	-0.94	14.75	26.35	-1.06	0.30
34	14.45	24.56	-0.55	13.07	24.90	-2.01	2.03
35	12.88	26.39	-0.55	13.12	26.38	-1.98	1.45
36	20.47	26.36	1.71	20.91	26.42	-0.19	1.96
49	20.65	32.76	-1.19	20.13	35.10	-0.05	2.65
50	20.72	35.92	-1.19	20.16	35.92	-0.10	1.22
51	12.80	36.09	-1.19	13.09	36.17	-0.73	0.55
52	12.78	35.26	-1.19	13.05	35.34	-0.86	0.43
55	13.16	33.77	-1.47	13.46	33.78	-1.23	0.38
56	13.16	33.01	-1.47	13.42	33.16	-1.43	0.30
57	19.01	33.74	-1.47	19.59	33.04	-0.93	1.05
58	19.01	32.97	-1.47	19.59	33.66	-1.05	0.99

different average numbers of incorrect correspondences for each noise level, respectively.

As shown in Figure 13, the algorithm works very well if the number of missing lines is 0, i.e. each 3D line is visible in all three images. For example, on average, there are only about 0.5 incorrect correspondences for the noise level of 5 pixels. For the lower levels of noise ranging from 1 pixel to 3 pixels, there is little effect on the performance of the algorithm for different numbers of missing points. For the higher levels of noise, the number of incorrect correspondences increases as the number of missing image lines increases. From Figure 13, we can see that on the average, the number of incorrect correspondences rises about 0.5 lines (about 2%) at any of the noise levels. Therefore, our experiments have shown that the algorithm can tolerate a difference in the number of image lines from three images and is robust against a reasonable level of noise.

6.2 PUMA Sequence

In this subsection, we test on the indoor PUMA image sequence again. For each image, 2D image line segments were extracted by the accurate Boldt line extraction algorithm [9]. Figures 14 shows a triplet of the extracted line sets from the 1*st*, 10*th*, and 20*th* frames in the sequence with 196, 185, and 189 image line segments, respectively. Here, the three images have more accurate image line segments, but also more line segments are extracted than in the previous set shown in Figure **??**. Figure **??** shows 76 correct image line segment correspondences. Again, this experiment uses the ground truth 3D data supplied in Kumar's thesis [49]. Here we only reported the comparisons between the reconstructed 3D lines and their ground truth data for those 3D lines whose ground truth coordinates of two endpoints are available. Table 3 and Table 4 report a comparison between the triangulated 3D lines and their ground-truth data. For the 35 line correspondences, the average orientation error is 3.36 degree, and the average distance error is 0.11 feet.

6.3 RADIUS Image Set

The goal of this experiment is to test the performance of correspondence process for larger size images and a huge line data set, and we will not attempt evaluation of 3D accuracy here. Again, this experiment uses the RADIUS image data set (J1-J8) supplied through the ARPA-ORD RADIUS project [4]. Each image contains approximately 1320×1035 pixels, with about 11 bits of grey level information per pixel. The dimensions of each image vary slightly because the images have been resampled, and unmodeled geometric and photometric distortions have been introduced to more accurately reflect actual operating conditions.

Here, the Boldt line algorithm [9], was run on all of the eight images J1-J8. To reduce the number of lines to a computationally manageable size these images were first reduced in resolution to half their original size before line extraction. After line extraction, the segments found were rescaled back into original image coordinates, then filtered so that each line segment in the final set has a length of at least 10 pixels and a contrast (difference in average grey level across the line) of at least 15 grey levels. This procedure produced more than 2000 line segments per image. Figures 16, 17, and 18 show three sets of line segments produced from images J1, J2, J3, respectively.

As shown in Figures 16, 17, and 18, the three line sets are huge, and the numbers of image line segments from the three images are different, since J1, J2, and J3 have 2662, 2772, and 2734 image line segments, respectively. Both the 2D image lines and the camera parameters are noisy. Clearly, there exists significant fragmentation in the three image line data sets. Due to this fragmentation, there may be several line segment correspondences that correspond to the same 3D line. Geometrically, each finite image line segment corresponds to a finite line segment in its corresponding 3D line. Due to fragmentation, each of three line segments in an image line correspondence often is from a different part of the actual 3D line triple. In order to reduce some unnecessary correspondences obtained by the line matching and reconstruction algorithm, a "common part" constraint was imposed. This constraint ensures that any line segment correspondence must have an overlapping common part in their 3D intersection line. Another advantage of this constraint is that it can eliminate some incorrect correspondences which have no common element in their corresponding 3D pseudo-intersection lines, although they have small affinity values. The result of the algorithm was 232 line segment correspondences across three images, shown in figures 19, 20, and 21.

7 Conclusions

This paper addresses the problems of determining image feature correspondences while simultaneously computing the corresponding 3D features, for images with known camera pose. Our novel contribution is the development and application of an affinity measure between image features (points and lines), i.e. a measure of the degree to which candidates from different images consistently represent the projection of the same 3D point or the same 3D line. We utilize optimal bipartite graph matching to solve the problem of simultaneous recovery of correspondence and 3D reconstruction. The matching mechanism is general and robust since it ensures that a maximal matching can be found based upon proven graph theoretical algorithms. From the point of view of implementation, this graph-based matching technique can be implemented efficiently and in parallel, and has been successfully applied to matching problems involving graphs of quite large size.

Experiments with both synthetic and real image data sets were conducted to evaluate performance of the point and line matching algorithms. The experiments have shown that the algorithms are robust in the presence of significant amounts of missing points and lines, and noise in the camera parameters and in the extracted image point and line features. The presented integrated matching and triangulation methods are well-suited for photogrammetric mapping applications where camera pose is already known, for wide-baseline multi-camera stereo systems, and for model extension where a set of known features are tracked. Also, these techniques potentially have a wider application domain than traditional matching and reconstruction algorithms, since our matching mechanism is general-purpose and only the affinity measures would need to be redefined. They could also be extended to deformable 3D matching and reconstruction problems.

Some remaining issues associated with generalization to multi-image analysis over larger numbers of images are subject for further study. In order to perform 3D reconstruction from m images, the point matching and triangulation algorithm could be repeated for each image pair of $\binom{m}{2}$, and the integrated line matching and triangulation algorithm could be repeated for each image triplet of $\binom{m}{3}$. This is not true multi-image matching, since all images are not used together, and the two affinity measures are not able to describe the affinity among image features (points and lines) over multiple images. The development of a new multi-image matching algorithm based on more general affinity measures is left for future work. Finally, it is desirable to develop a unified matching and reconstruction algorithm based on both image points and image lines, combining the advantages of both.

References

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

[2] Aggarwal, J. K., Davis, L. S. and Martin, W. N., Correspondence processes in dynamic scene analysis. In *Proc. of IEEE 69*, 1981.

[3] Aloimonos, J., and Rigoutsos, I., Determining 3-D motion of a rigid surface patch without correspondences under perspective projection. In *Proc. of AAAI*, August 1986.

[4] ARPA, RADIUS, Section III, In *Arpa Image Understanding Workshop.* Monterey, CA, 1994.

[5] Ayache, N., Artificial vision for mobile robots: stereo vision and multisensory perception. *The MIT Press*, 1991.

[6] Basu, A., and Aloimonos J., A robust algorithm for determining the translation of a rigidly moving surface without correspondence for robotics applications. In *Proc. of IJCAI*, August 1987.

[7] Bedekar, A. S., and Haralick, R. M., Finding corresponding points based on Bayesian triangulation. In *Proc. of CVPR'96*, San Francisco, CA, 1996.

[8] Blostein, S. D., and Huang, T. S., Quantization errors in stereo triangulation. In *Proc. of First International Conf. on Computer Vsion*, 1987.

[9] Boldt, M., Weiss, R., and Riseman, E. M., Token-based extraction of straight lines. *IEEE Trans. SMC*, Vol. 19(6), 1989.

[10] Burns, J. B., Hanson, A. R., and Riseman, E. M., Extracting straight lines. *IEEE Trans. PAMI*, Vol. 8(4), July 1986.

[11] Chang, S. F., and McCormick, S. T., A fast implementation of a bipartite matching algorithm, *Technical Report, Columbia University*, New York, 1989.

[12] Cheng, Y. Q., Acquisition of 3D Models from a Set of 2D Images. *Ph.D. Thesis*, CMPSCI TR97-04, Computer Science Department, University of Massachusetts, February 1997.

[13] Cheng, Y. Q., Collins, R., Hanson, A. R., and Riseman, E. M., Triangulation without correspondences. *Arpa Image Understanding Workshop,* Monterey, CA, 1994.

[14] Cheng, Y. Q., Wu, V., Collins, R. T., Hanson, A. R. and Riseman, E. M. Maximum-weight bipartite matching technique and its application in image feature matching. In *Proc. SPIE VCIP*, 1996.

[15] Cheriyan, J., and Maheshwari, S. N., Analysis of preflow push algorithms for maximum network flow. *SIAM J. Comput.* 18, 1989.

[16] Collins, R. T., A space-sweep approach to true multi-image matching. In *Proc. of CVPR'96*, San Francisco, CA, 1996.

[17] Collins, R.T., Cheng, Y.Q., Jayes, C., Stolle, F., Wang, X.G., Hanson, A.R., and Riseman, E.M., The ASCENDER system automated site modeling from multiple aerial images. *Computer Vision and Image Understanding*, Vol 72(2), pp. 143-162, 1998.

[18] Collins, R. T., Cheng, Y. Q., Jayes, C., Stolle, F., Wang, X.G., Hanson, A. R., and Riseman, E. M., Site model acquisition and extension from aerial images. In *Proc. ICCV* IEEE, 1996.

[19] Collins, R. T., Hanson, A. R., Riseman, E. M., and Cheng, Y. Q., Model matching and extension for automated 3D modeling. In *Proc. IUW*, 1993.

[20] Collins, R. T., Hanson, A. R., Riseman, E. M., Jayes, C. O., Stolle, F., Wang, X. G., and Cheng, Y. Q., UMass progress in 3D building model acquisition. In *Arpa Image Understanding Workshop*, Palm Springs, CA, Feb. 1996.

[21] Collins, R. T., Jayes, C. O., Stolle, F., Wang, X. G., Cheng, Y. Q., Hanson, A. R., Riseman, E. M., A system for automated site model acquisition. In *Integrating Photogrammetric Techniques with Scene Analysis and Machine Vision II*, SPIE Proceeding. Vol. 7617, Orlando, FL, April, 1995.

[22] Chou, T. C., and Kanatani, K., Recovering 3D rigid motions without correspondences. In *Proc. ICCV*, June 1987.

[23] Deriche, R., Vaillant, R., and Faugeras, O. D., From noisy edge points to 3D reconstruction of a scene: a robust approach and its uncertainty analysis. In *Proc. of the Second Europeian Conf. on Computer Vision*, 1992.

[24] Edmonds, J., Maximum matching and polyhedron with 0,1 vertices. *J. Research of the National Bureau of Standards*, Vol. 69B, 1965.

[25] Edmonds, J., and Karp, R. M. Karp, Theoretical improvements in algorithmic efficiency for network flow problems. *J. Assoc. Comput. Math.*, Vol 19, 1972.

[26] Ford, L. R., and Fulkerson, E., Flows in networks. *Princeton Univ. Press*, Princeton NJ, 1962.

[27] Gabow, H. N., A scaling algorithm for weighted matching on general graphs. In *Proc. 26th Annual Symp. of the Foundations of Computer Science*, 1985.

[28] Gabow, H. N., Data structure for weighted matching and nearest common ancestors with linking. 1990.

[29] Ganapathy, S., Decomposition of transformation matrices for robot vision. In *Proc. IEEE Int. Conf. Robotics and Automation*, Atlanta, GA, Mar. 1984.

[30] Gold, S., and Rangarajan, A., Graph matching by graduated assignment. In *Proc. of CVPR'96*, San Francisco, CA, 1996.

[31] Goldberg, A. V., and Tarjan, R. E., A new approach to the maximum-flow problem. *J. Assoc. Comput. Mach.*, Vol. 35, 1988.

[32] Goldberg, A. V., Efficient graph algorithms for sequential and parallel computers. *Ph.D thesis*, MIT, Cambridge, Mass, January, 1987.

[33] Goldgof, D. B., Lee, H., and Huang, T. S., Matching and motion estimation of three-dimensional point and line sets using eigenstructure without correspondences. *Pattern Recognition*, Vol.25, No. 3, 1992.

[34] Griffin, P.M., Correspondence of 2-D projections by bipartite matching. *Pattern Recognition Letter*, No. 9, pp.361-366, 1989.

[35] Gruen, A., and Baltsavias, E., Geometrically constrained multiphoto matching. *Photogrammetric Engineering and Remote Sensing*, Vol. 54(6), 1988, pp.633-641.

[36] Hao, J., and Kocur, G., An implementation of a shortest augmenting path algorithm for the assignment problem. *Network flows and matching: first DIMACS implementation challenge, Series in DIMACS*, Vol. 12, pp. 453-468, 1993.

[37] Hartley, R. I., Triangulation. In *Proc. IUW*, 1994.

[38] Hopcroft, J. E., and Karp, R. M., An $n^{2.5}$ algorithm for maximum matching in bipartite graphs. *J. ACM*, Vol. 23, pp. 225-231, 1973.

[39] Ito, E., and Aloimonos, J., Is correspondence necessary for the perception of structure from motion?. In *Image Understanding Workshop*, pp. 921-929, April 1988.

[40] Jaynes, C., Stoll, F., and Collins, R., Task driven perceptual organization for extraction of rooftop polygons. In *Proceeding ARPA Image Understanding Workshop*, Monterey, CA, Nov. 1994.

[41] Johnson, D. S., and McGeoch, C. C., Network flows and matching: first DIMACS implementation challenge. *Series in DIMACS*, Vol. 12, 1993.

[42] Jonker, R., and Volgenant, A., A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, Vol. 38, pp. 325-340, 1987.

[43] Kim, W., and Kak, A., 3D object recognition using bipartite matching embedded in discrete relaxation. *IEEE PAMI*, 13(3), pp. 224-251, March 1991.

[44] Kahn, P., Kitchen, L., and Riseman, E., Real-time feature extraction: a fast line finder for vision-guided robot navigation. *IEEE Pattern Analysis and machine Intelligence*, Vol.12, No.11, pp.1098-1102, Nov. 1990.

[45] Kuhn, H. W., The Hungarian method for assignment problem. *Naval Research Logistics*, Quarterly 2, pp. 83-97, 1955.

[46] Kumar, R., and Hanson, A., Model Extension and Refinement Using Pose Recovery Techniques. *Journal of Robotic Systems*, 9(6):753-771, 1992.

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

[48] Kumar, R., Anandan, P., and Hanna, K., Shape recovery from multiple views: a parallax based approach. In *Proc. of ARPA Image Understanding*, Monterey, CA Nov. 1994.

[49] Kumar, R. Model Dependent Inference of 3D Information from a Sequence of 2D Images. *Ph.D. Thesis*, CMPSCI TR92-04, Computer Science Department, University of Massachusetts, February 1992.

[50] Lee, H., Lin, Z., and Huang, T. S., Finding 3-D point correspondences in motion estimation. In *Proc. of Eighth Int. Conf. on Pattern Recognition*, pp. 303-305, 1986.

[51] Lee, C-H., and Joshi, A., Correspondence problem in image sequence analysis. *Pattern Recognition*, Vol. 26, No. 1, pp. 47-61, 1993.

[52] Lee, H., Lin, Z., and Huang, T. S., Estimating rigid-body motion from three-dimentional data without matching point correspondences. *Int. J. Imaging Systems Technol.* Vol. 2, pp. 55-62, 1990.

[53] Lessard, R., Rousseau, J.-M., and Minoux, M., A new algorithm for general matching problems using network flow subproblems. *Networks*, Vol. 19, pp. 459-479, 1989.

[54] Martin Marietta and SRI International, RCDE User's Guide. *Martin Marietta, Management and Data Systems*, Philadelpha, PA, 1993.

[55] Micali, S., and Vazirani, V., An $O(\sqrt{\mid V \mid} \cdot \mid E \mid)$ algorithm for finding maximum matchings in general graphs. In *Proc. 21st Symp. Foundations of Computer Science*, pp. 17-27, 1980.

[56] Pentland, A., and Sclaroff, S., Closed-form solutions for physically-based shape modeling and recognition. *IEEE PAMI*, 13(7), pp. 715-729, July 1991.

[57] Pentland, A., and Horowitz, B., Recovery of non-rigid motion and structure. *IEEE PAMI*, 13(7), pp. 730-742, July 1991.

[58] Roy, S. and Cox, I.J., A maximum-flow formulation of the N-camera stereo correspondence problem. *ICCV98*, pp. 492-499, 1998.

[59] Sclaroff, S., and Pentland, A., A modal framework for correspondence and description. In *Proc. of IEEE*, pp. 308-313, 1993.

[60] Scott, G. L., and Longuet-Higgins, H. C., An algorithm for associating the features of two patterns. In *Proc. Roy Soc Lond*, Vol. B244, pp. 21-26, 1991.

[61] Shapiro, L. S., and Brady, J. M., Feature-based correspondence: an eigenvector approach. *Image and Vision Computing*, Vol. 10, No. 5, pp. 283-288, June, 1992.

[62] Ullman, S., The interpretation of visual motion. *MIT Press*, 1979.

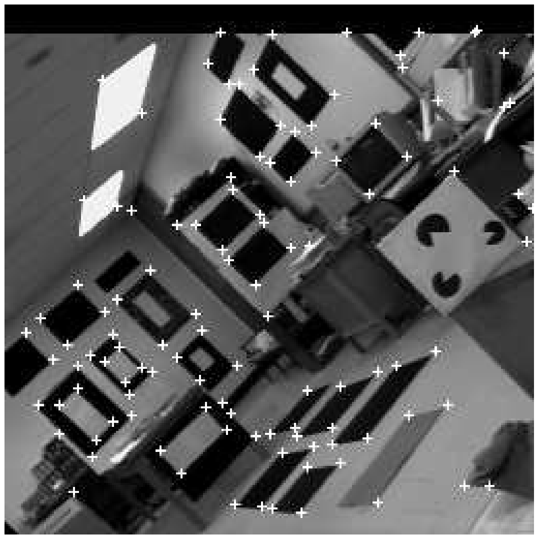
[63] Wang, X., Cheng, Y. Q., Collins, R. T., and Hanson, A. R., Determining correspondences and rigid motion of 3-D point sets with missing data. In *Proc. CVPR* IEEE, 1996.

[64] Wu, M.S. and Leou, J.J., A bipartite matching approach to feature correspondence in stereo vision. *Pattern Recognition Letter*, No.16, pp. 23-31, 1995.

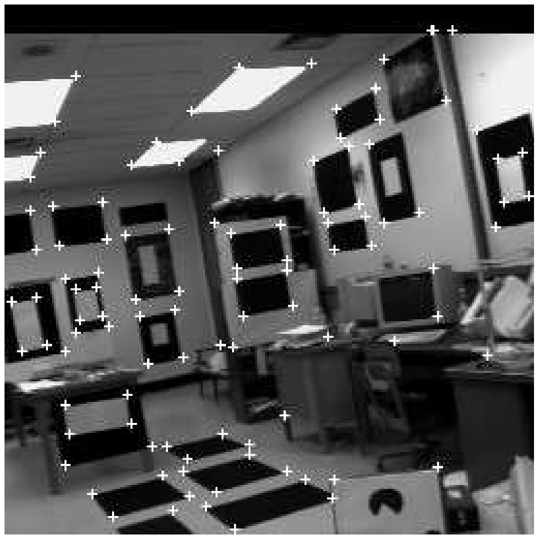
[65] Wu, Z., and Leahy, R., An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *IEEE PAMI*, 13(7), pp. 1101-1113, Nov. 1993.

[66] Zhang, Z., and Faugeras, O. D., Building a 3D world model with a mobile robot: 3D line segment representation and integration. In *Proc. of the IEEE International Conf. on Pattern Recognition*, Atlantic City, New Jersy, June, 1990.

[67] Zhang, Z., and Faugeras, O. D., Tracking and grouping 3D line segements. In *Proc. of 3rd International Conf. on Computer Vision*, pp. 577-580, Osaka, Japan, 1990.

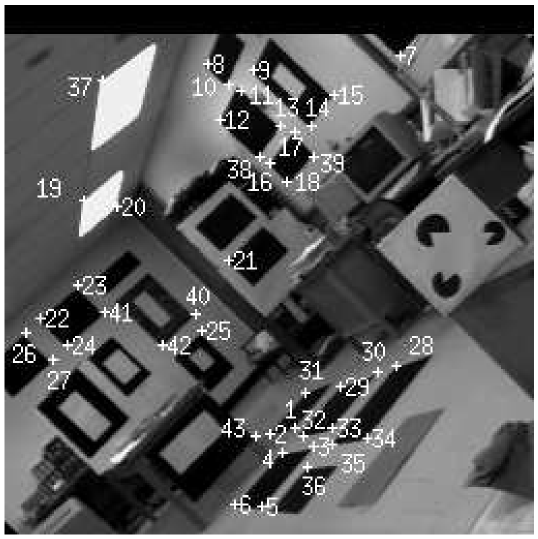


(a)

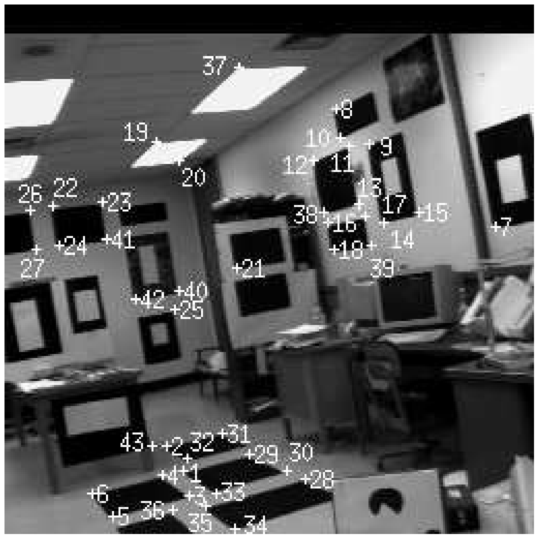


(b)

Figure 7: PUMA sequence data for matching experiments: (a) extracted image points (113 points) in the $1st$ frame; (b) extracted image points (107 points) in the $10th$ frame.



(a)



(b)

Figure 8: (a) 43 matched image points in the $1st$ frame; (b) 43 matched image points in the $10th$ frame.

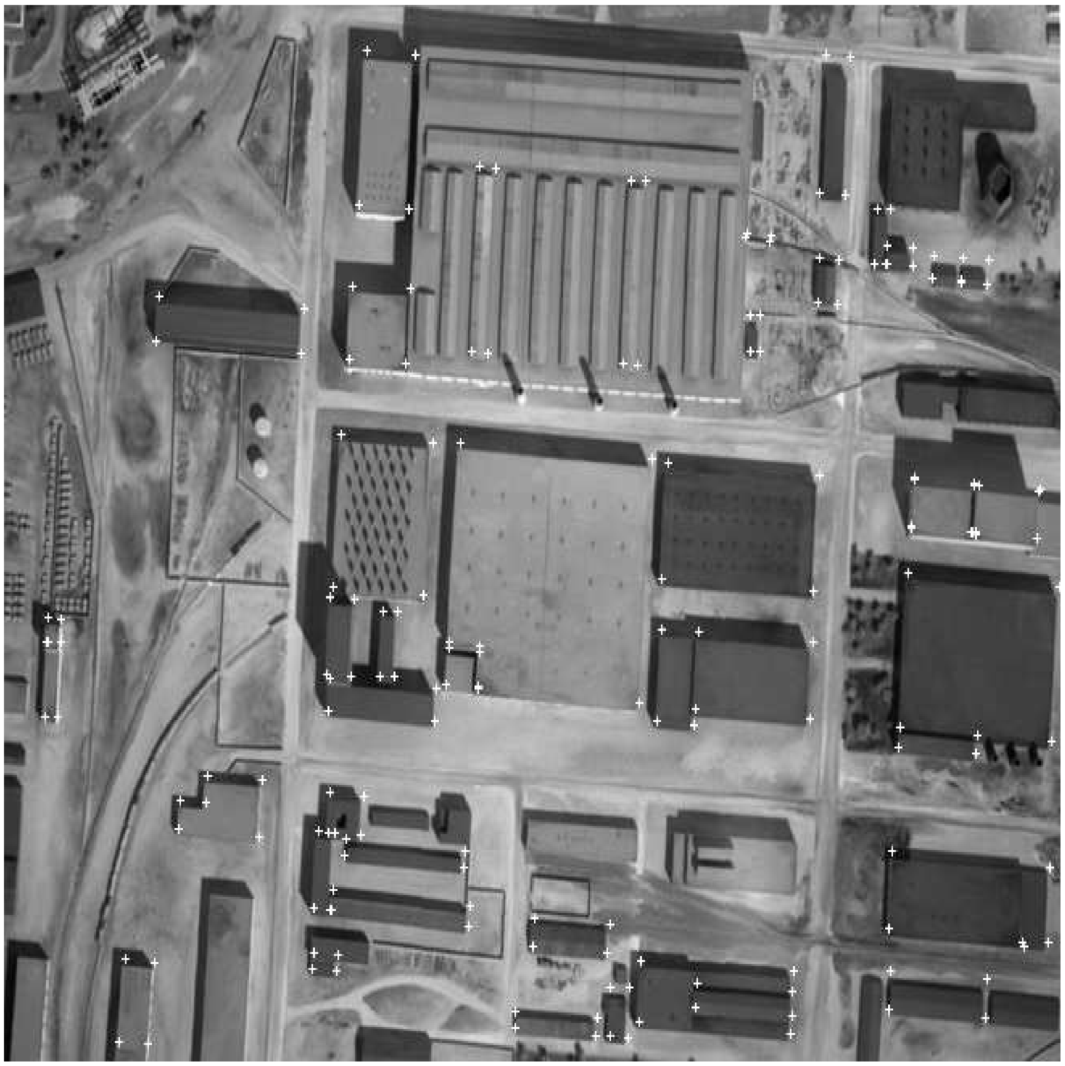


Figure 9: Extracted image points in image J3. The 186 points are a result of generating 2D building polygons via the ASCENDER system.

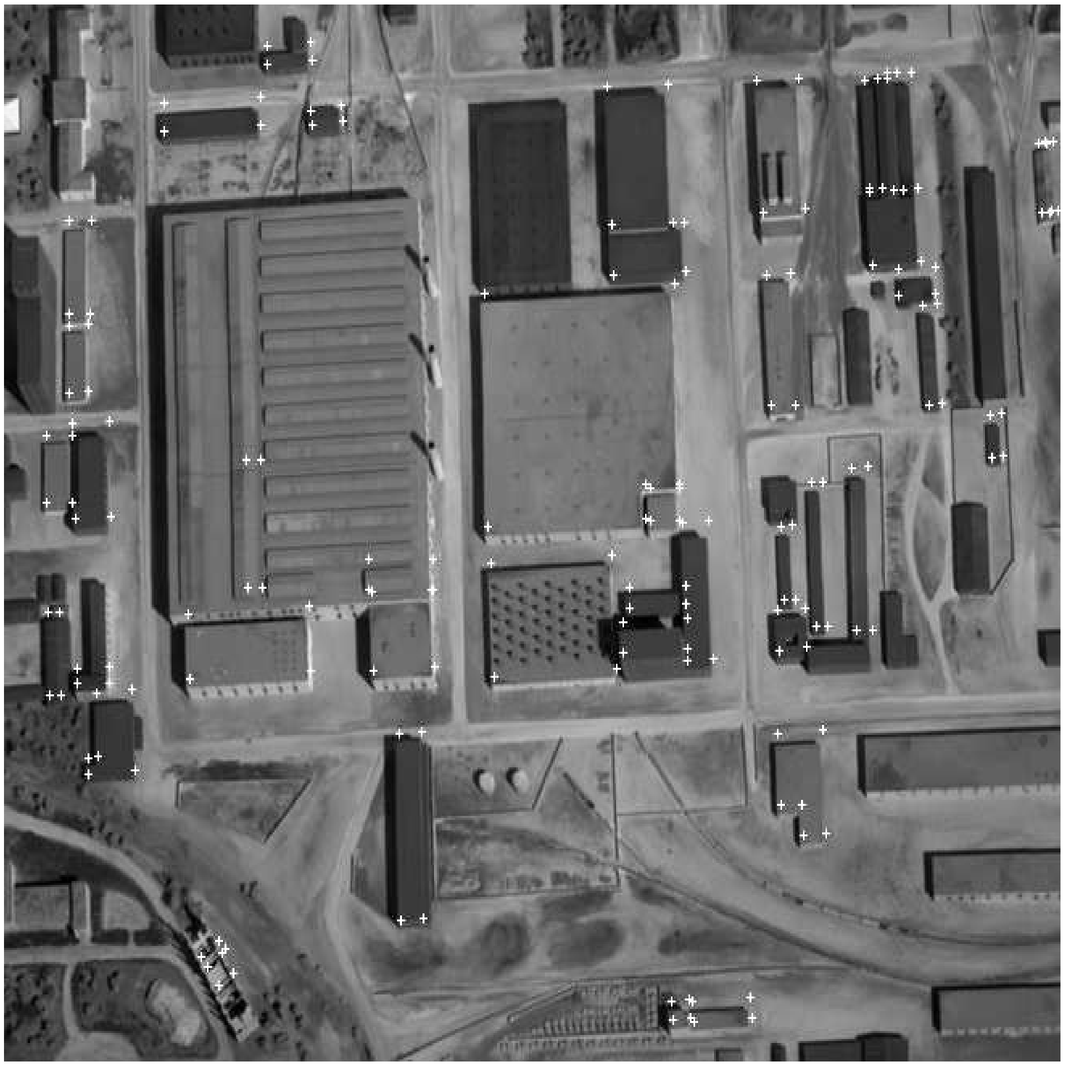


Figure 10: Extracted image points (186 points) in image J7.

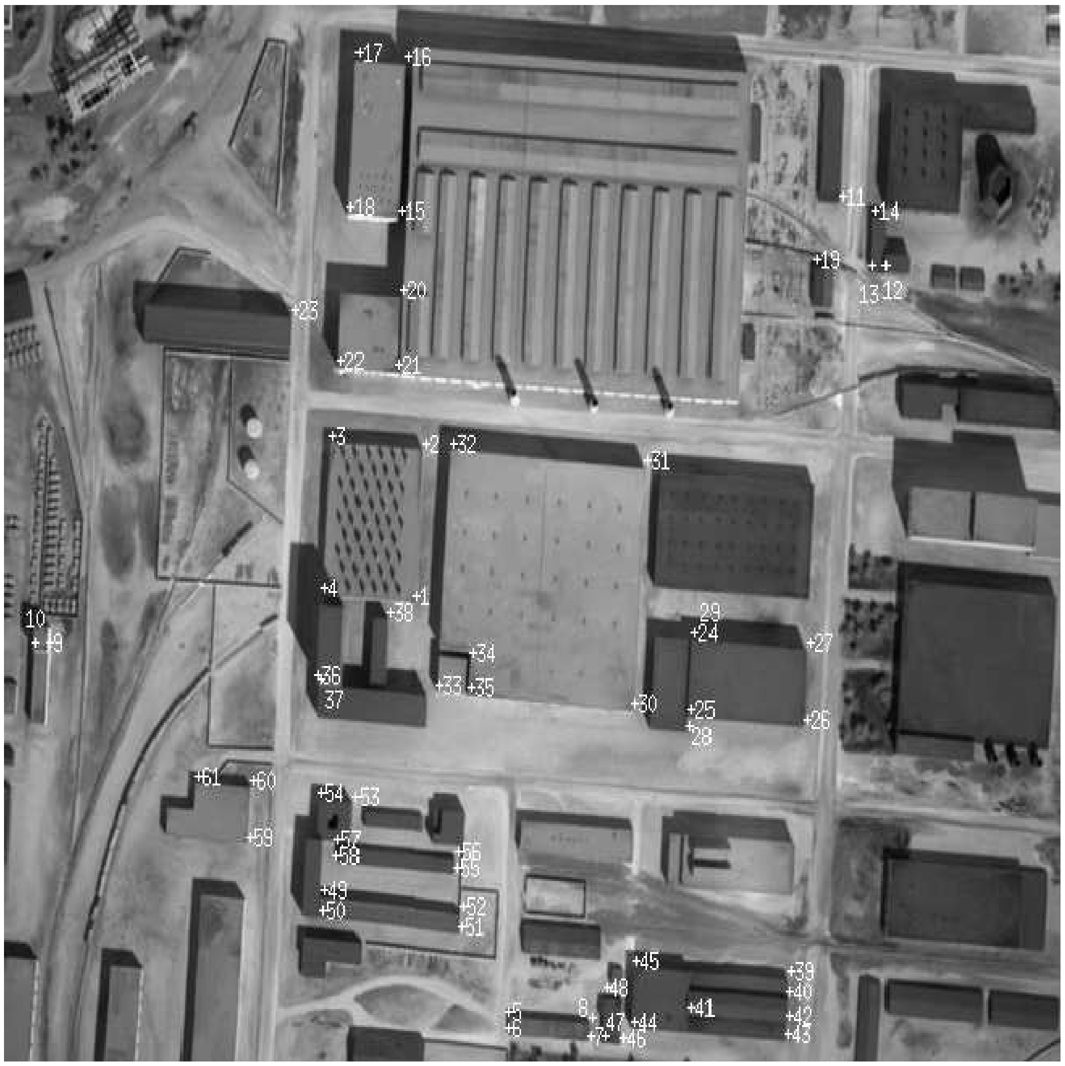


Figure 11: Experiments with RADIUS detect 61 matched image points in image J3.

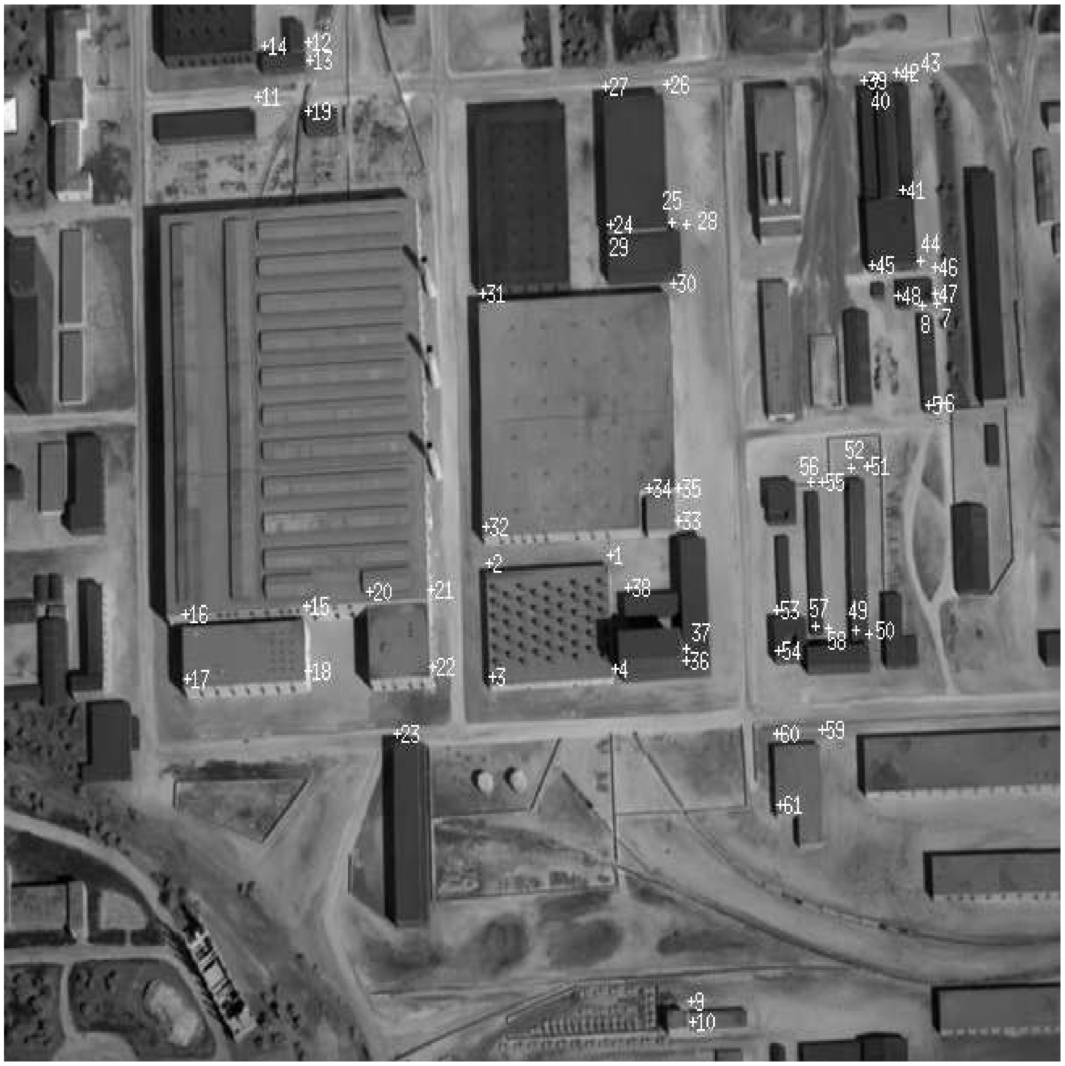


Figure 12: Experiments with RADIUS detect 61 matched image points in image J7.

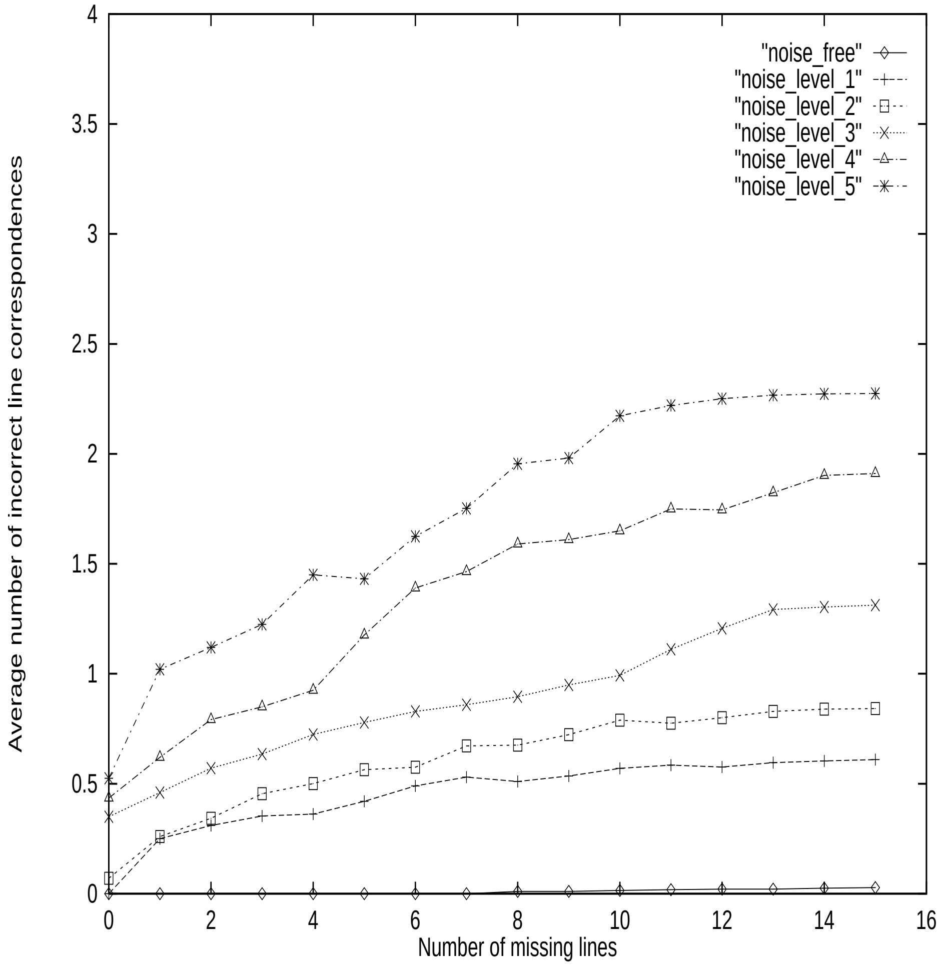
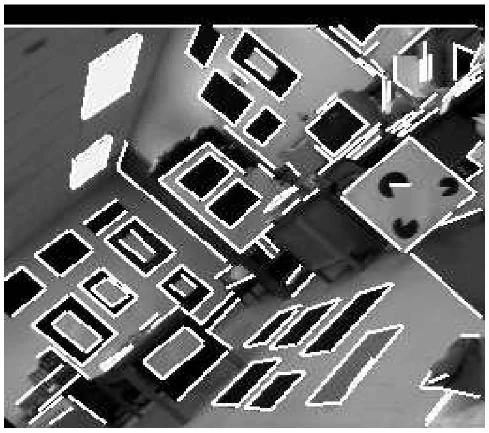
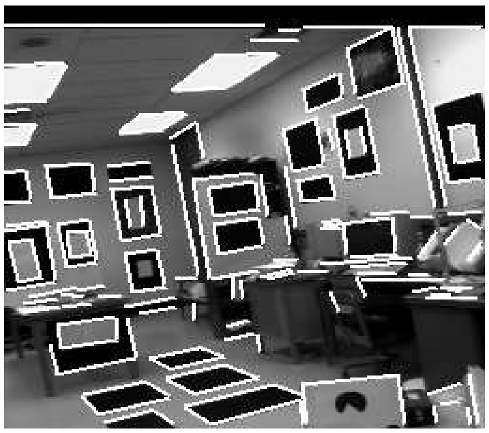


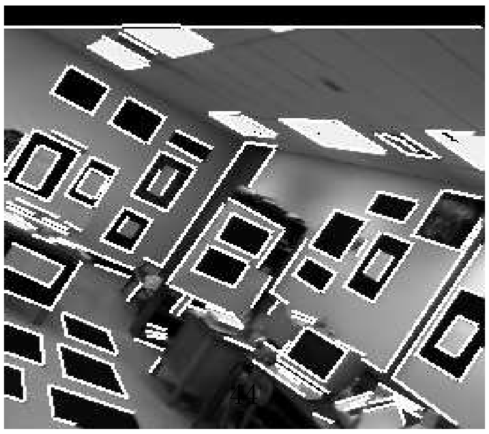
Figure 13: Performance of the image line matching algorithm against 5 levels of noise with different numbers of missing points.



(a)

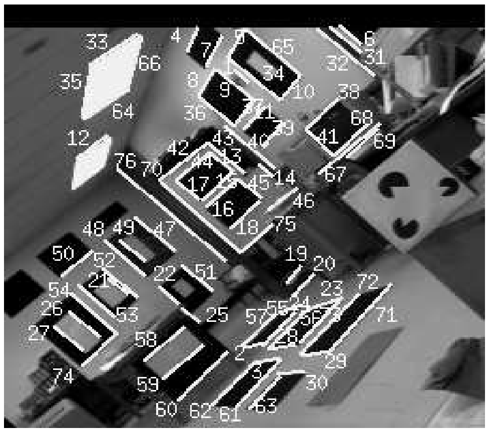


(b)

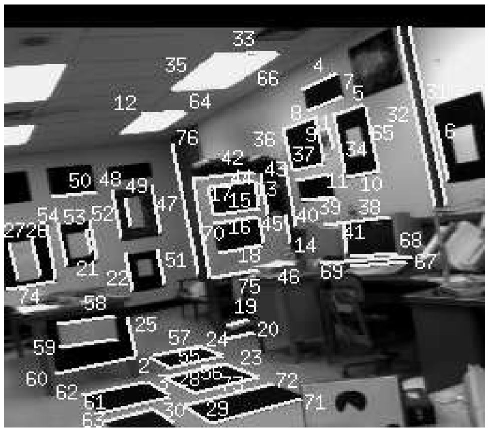


(c)

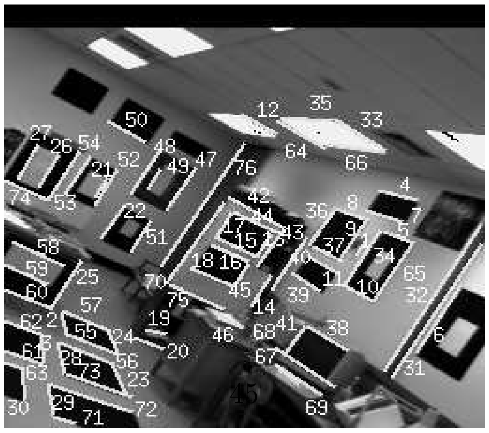
Figure 14: Boldt lines in PUMA sequence. Three sets of image line segments extracted by the Boldt algorithm: (a) 196 image line segments in frame 1; (b) 185 image line segments in frame 10; (c) 189 image line segments in frame 20.



(a)



(b)



(c)

Figure 15: 76 matched line segments from Boldt lines in PUMA sequence: (a) frame 1; (b) frame 10; (c) frame 20.

Table 3: 3D line reconstruction error for the PUMA Sequence frames using Boldt line algorithm: Processing of Frames 1,10,20 produces an average orientation error of 3.36 degree, and average distance error of 0.11 feet (Part 1).

Line	Actual 3D Lines			Computed 3D Lines			Orient. Error	Distance Error
	u_x	u_y	u_z	u_x	u_y	u_z		
	m_x	m_y	m_z	m_x	m_y	m_z		
4	-0.00	-0.00	1.00	0.08	0.02	1.00		
	-8.19	-0.00	-0.00	-7.98	-0.91	0.66	4.74	0.05
5	-0.00	-0.01	1.00	0.00	-0.02	1.00		
	-7.01	-0.00	-0.00	-7.13	0.16	0.03	0.21	0.19
7	0.00	0.00	1.00	0.01	0.01	1.00		
	-7.25	0.00	0.00	-7.30	0.04	0.09	0.71	0.05
8	-0.00	-0.00	1.00	0.02	-0.00	1.00		
	-7.03	-0.00	-0.00	-7.15	-0.09	0.17	1.38	0.12
9	0.00	1.00	0.00	0.12	0.97	-0.20		
	13.89	0.00	0.00	14.50	-1.65	0.62	13.41	0.01
10	0.00	-0.02	1.00	-0.01	-0.01	1.00		
	-5.02	0.00	0.00	-4.84	0.35	-0.04	0.95	0.22
11	-0.00	-0.03	1.00	0.02	-0.04	1.00		
	-5.34	-0.00	-0.00	-5.49	-0.09	0.12	1.39	0.12
12	1.00	-0.00	-0.00	1.00	-0.02	0.10		
	-0.00	-7.03	9.01	-1.05	-6.71	9.10	5.90	0.09
15	1.00	0.01	0.00	1.00	0.00	0.06		
	0.14	-11.28	4.17	-0.23	-11.09	4.16	3.60	0.03
17	0.00	1.00	0.00	0.06	0.99	-0.16		
	11.28	0.00	2.94	12.63	-0.09	3.44	9.58	0.00
18	1.00	0.01	0.00	1.00	0.01	-0.04		
	0.14	-11.28	2.79	0.16	-10.87	2.74	2.09	0.02
23	-0.00	-0.00	1.00	0.01	0.00	1.00		
	0.47	-4.82	-0.00	0.51	-4.90	0.00	0.63	0.04
24	-0.00	-0.00	1.00	-0.03	0.02	1.00		
	0.47	-4.43	-0.00	0.72	-3.95	0.09	1.93	0.02
29	0.00	0.00	1.00	0.02	-0.02	1.00		
	0.47	-6.94	0.00	0.09	-7.35	-0.16	1.91	0.01
30	-0.00	-0.00	1.00	0.01	-0.00	1.00		
	0.47	-7.53	-0.00	0.38	-7.63	-0.03	0.41	0.02
33	1.00	-0.00	-0.00	1.00	0.00	-0.04		
	-0.00	-15.03	9.01	0.45	-14.65	9.14	2.48	0.10
34	0.00	1.00	0.01	0.02	1.00	-0.04		
	14.85	0.00	0.00	15.24	-0.31	0.20	3.31	0.03
35	0.00	0.00	1.00	0.00	0.01	1.00		
	-9.01	-3.32	0.00	-9.05	-3.34	0.07	0.49	0.04

Table 4: 3D line reconstruction error for the PUMA Sequence frames using Boldt line algorithm: Processing of Frames 1,10,20 produces an average orientation error of 3.36 degree, and average distance error of 0.11 feet (Part 2)(continue from Table 3).

Line	Actual 3D Lines			Computed 3D Lines			Orient. Error	Distance Error
	u_x	u_y	u_z	u_x	u_y	u_z		
	m_x	m_y	m_z	m_x	m_y	m_z		
36	0.00	1.00	-0.00	-0.10	0.98	0.18		
	11.78	0.00	0.00	10.56	1.21	-0.54	11.93	0.01
37	0.00	-0.00	1.00	-0.00	-0.02	1.00		
	-5.40	0.00	0.00	-5.65	0.26	-0.00	0.90	0.22
39	0.00	-0.07	1.00	0.00	-0.03	1.00		
	-5.05	0.00	0.00	-4.55	0.19	0.02	2.17	0.21
44	1.00	0.02	-0.00	1.00	0.01	-0.02		
	0.28	-11.28	5.26	0.21	-10.99	5.28	1.45	0.19
50	1.00	-0.02	0.00	0.99	-0.03	0.13		
	0.00	0.00	6.37	-0.81	0.15	6.35	7.34	0.01
56	1.00	0.00	0.01	1.00	-0.01	0.01		
	0.00	-14.85	-0.47	-0.19	-14.59	-0.60	0.74	0.25
58	1.00	0.01	0.00	1.00	-0.06	0.01		
	0.19	-13.75	1.57	-0.83	-13.51	1.00	4.27	0.31
59	1.00	-0.05	0.00	1.00	-0.05	0.06		
	-0.74	-13.73	0.37	-0.77	-13.85	0.32	3.44	0.04
60	1.00	-0.00	0.00	1.00	-0.00	0.07		
	-0.06	-13.75	0.08	-0.08	-13.91	0.02	3.95	0.06
61	1.00	0.00	0.00	1.00	0.01	0.08		
	0.00	-16.81	-0.47	0.20	-17.14	-0.43	4.74	0.01
63	1.00	0.00	0.05	1.00	-0.00	0.03		
	0.02	-17.92	-0.47	-0.08	-17.52	-0.54	1.25	0.02
64	1.00	0.00	0.00	1.00	-0.02	0.06		
	0.00	-11.13	9.01	-0.71	-10.82	9.06	3.48	0.03
65	0.00	1.00	0.00	-0.10	0.99	0.13		
	16.00	0.00	0.00	15.73	1.67	-0.47	9.26	0.34
66	0.00	0.00	1.00	-0.00	0.01	1.00		
	-9.01	-1.45	0.00	-9.11	-1.04	-0.01	0.47	0.42
71	1.00	0.00	0.00	1.00	-0.01	-0.05		
	0.00	-19.48	-0.47	-0.31	-18.98	-0.59	3.20	0.01
72	1.00	0.00	0.00	1.00	-0.01	-0.01		
	0.00	-17.82	-0.47	-0.15	-17.59	-0.57	0.65	0.12
73	1.00	-0.00	-0.00	1.00	-0.00	-0.00		
	-0.00	-16.95	-0.47	-0.08	-16.73	-0.54	0.29	0.20

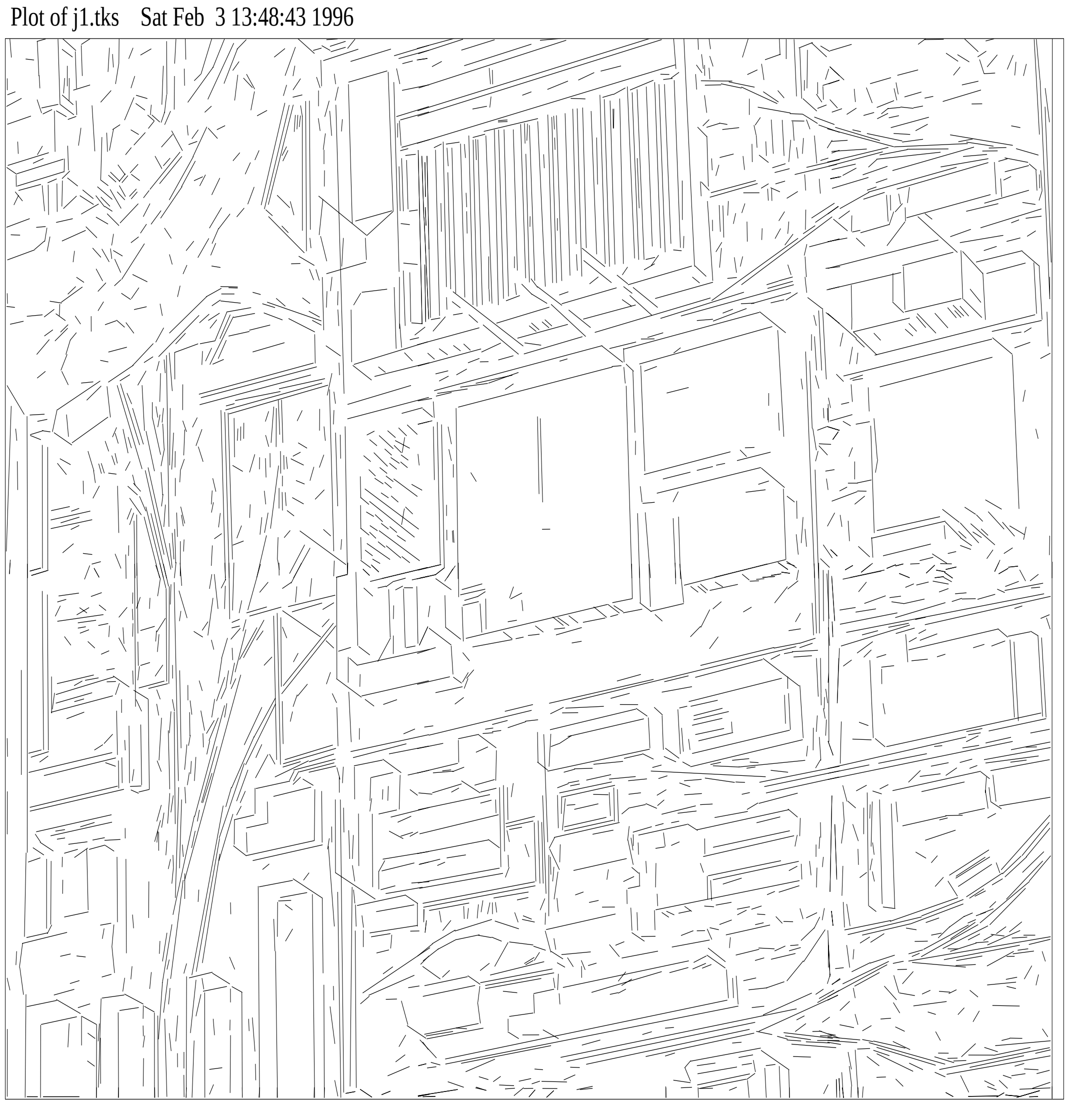


Figure 16: RADIUS model board image J1. The Boldt straight line extraction algorithm produced 2662 lines.

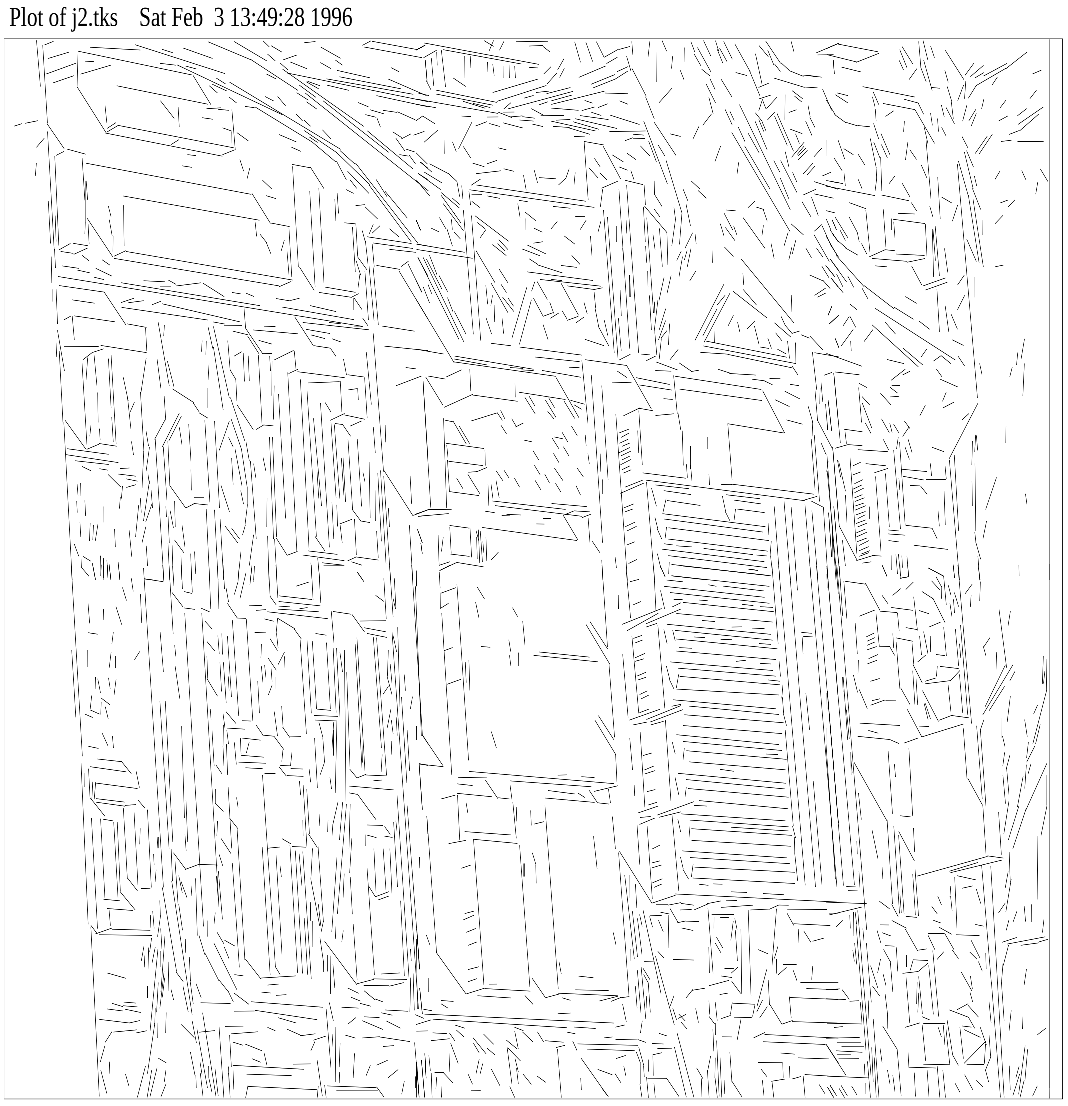


Figure 17: RADIUS model board image J2. The Boldt straight line extraction algorithm produced 2772 lines.

Plot of j3.tks Sat Feb 3 13:50:00 1996

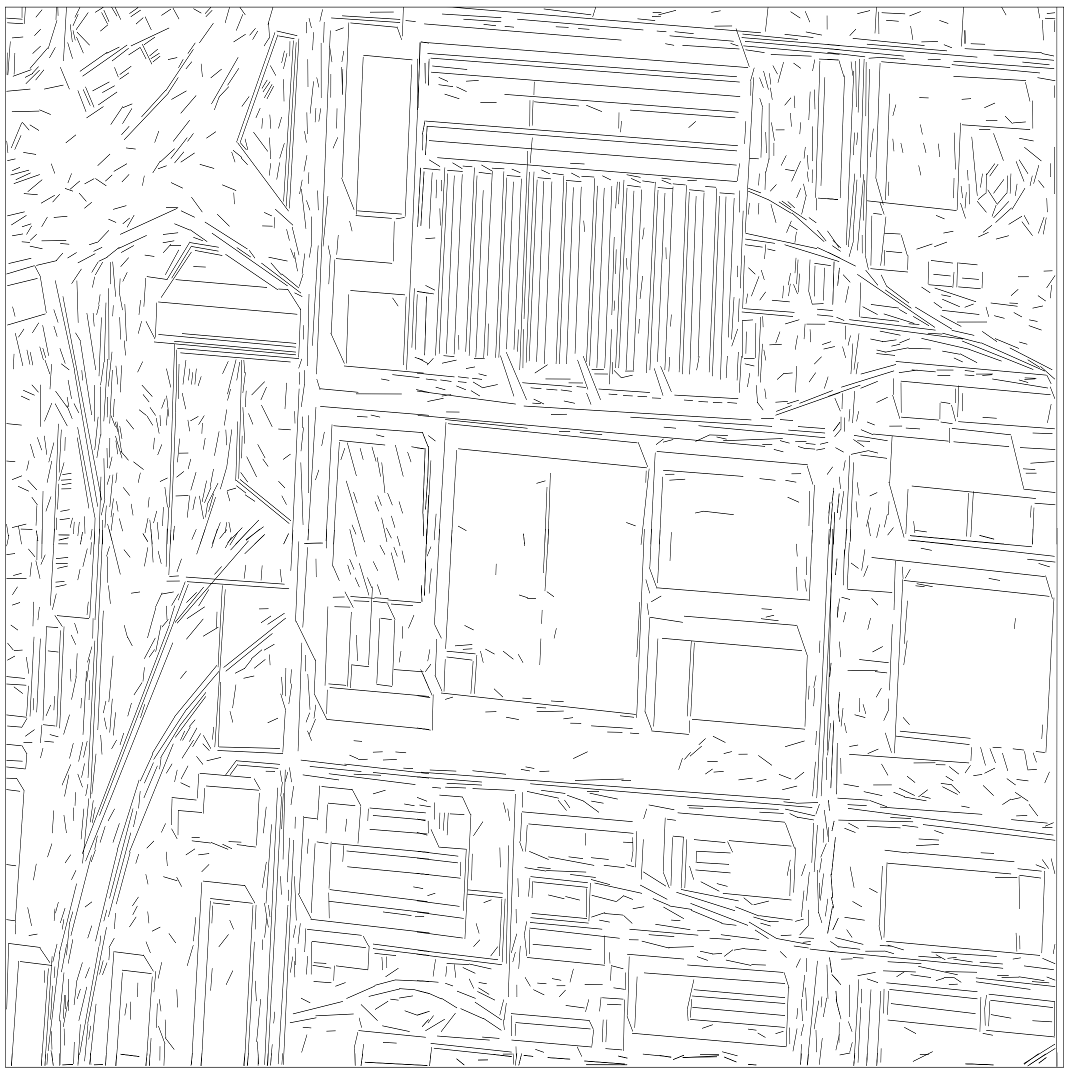


Figure 18: RADIUS model board image J3. The Boldt straight line extraction algorithm produced 2734 lines.

Plot of j1.tks   Tue Jan 23 20:09:30 1996

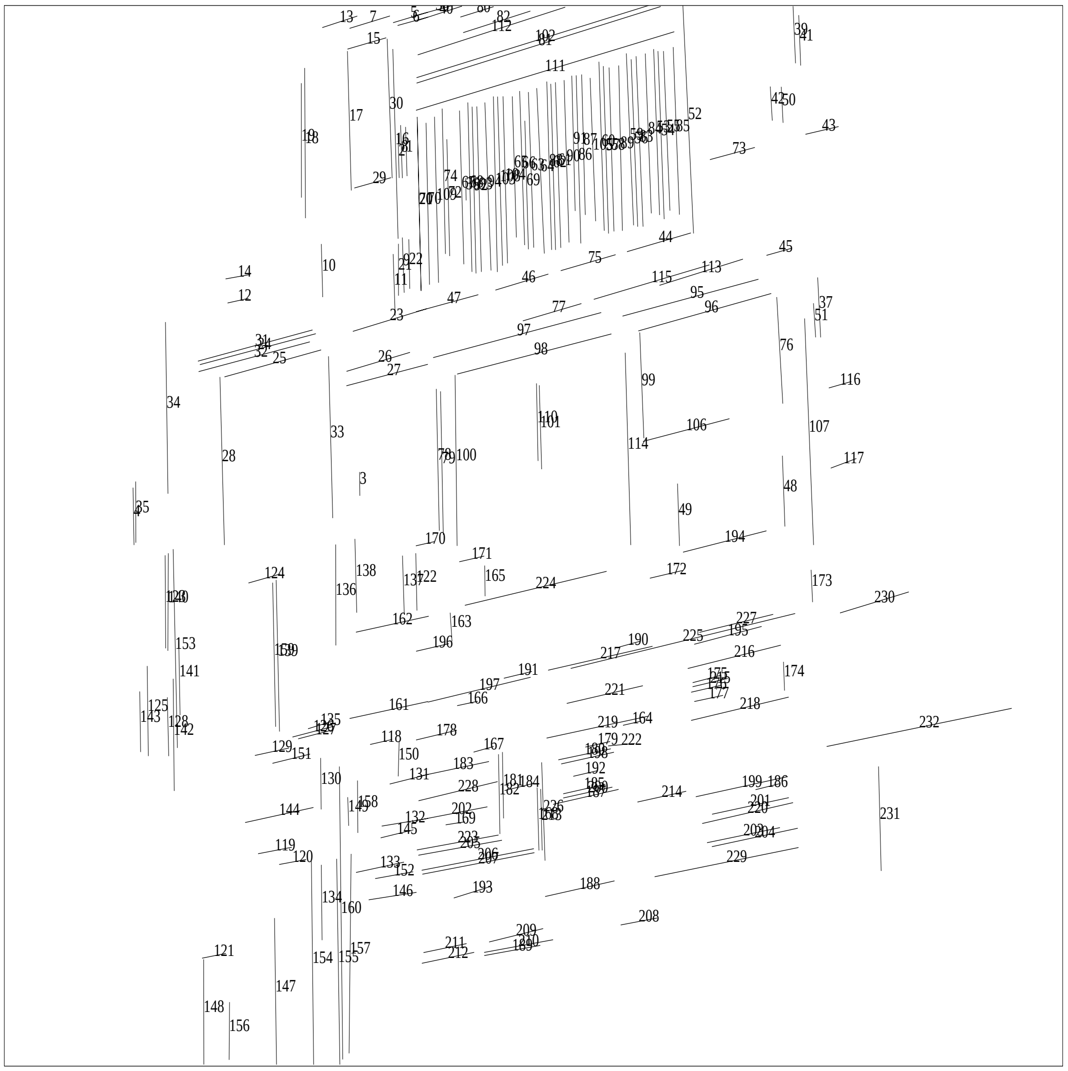


Figure 19: 232 matched line segments for RADIUS model board image J1.

Plot of j2.tks   Tue Jan 23 20:10:16 1996

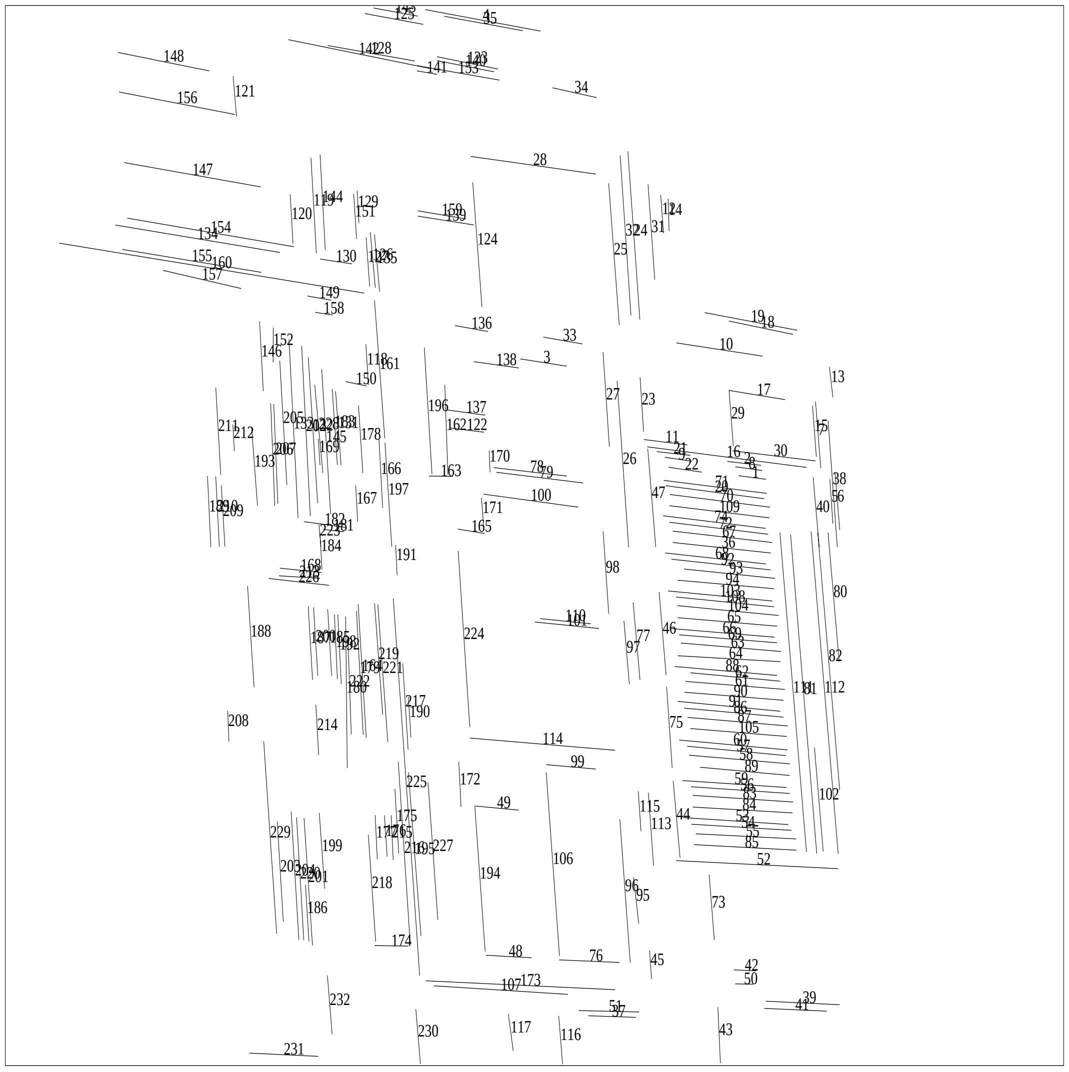


Figure 20: 232 matched line segments for RADIUS model board image J2.

Plot of j3.tks   Tue Jan 23 20:10:16 1996

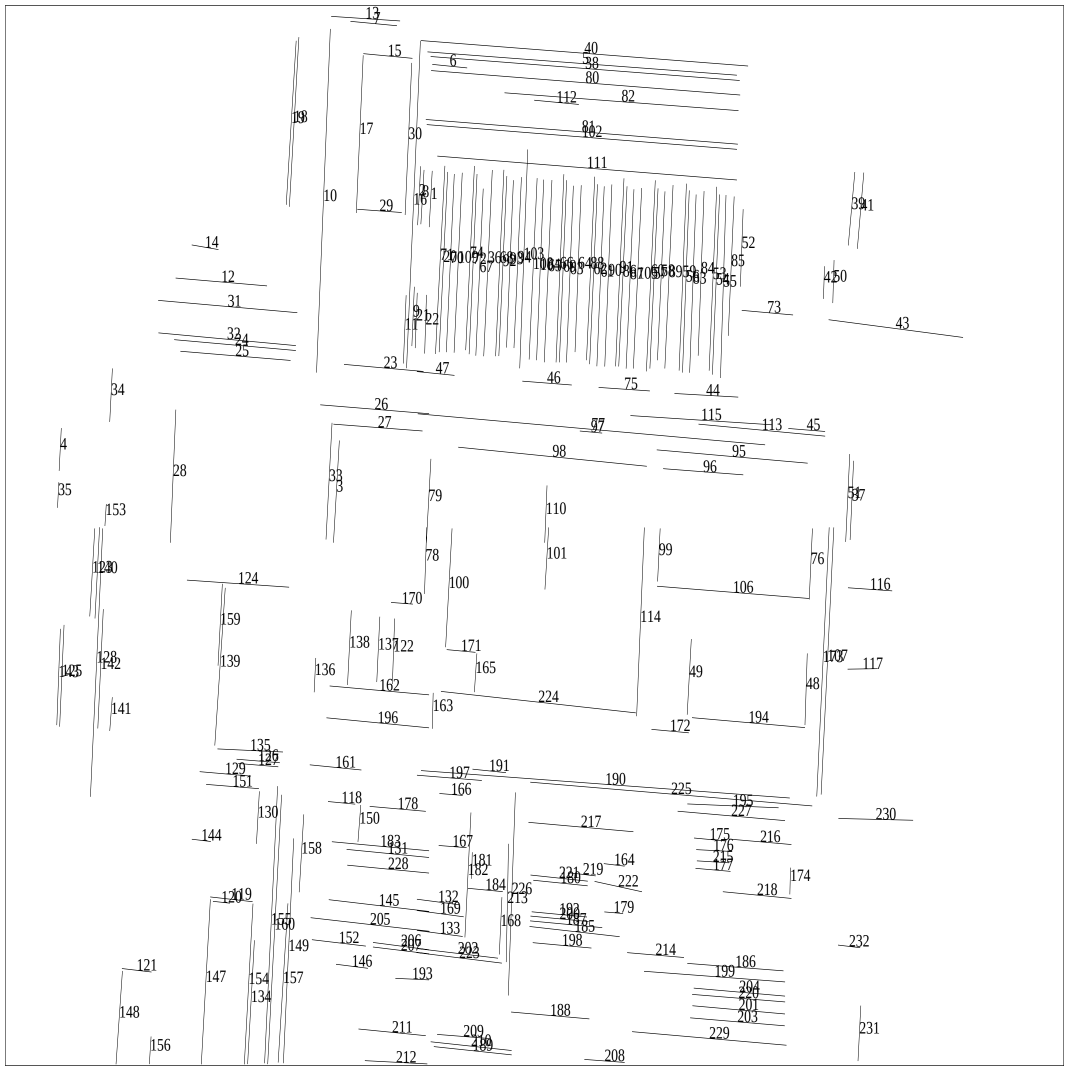


Figure 21: 232 matched line segments for RADIUS model board image J3.