Planar Detection Using Modified Expectation Maximization

A Thesis
presented to
the Faculty of the Graduate School
University of Missouri-Columbia

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science

by

Daniel Conrad

Dr. Guilherme DeSouza, Thesis Advisor

May 2011

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

PLANAR DETECTION USING MODIFIED EXPECTATION MAXIMIZATION

Presented by Daniel Conrad,
a candidate for the degree of Master of Science,
and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Guilherme N. DeSouza, Assistant Professor, Dept. of Electrical and Computer Engineering

---

Dr. Michela Becchi, Assistant Professor, Dept. of Electrical and Computer Engineering

---

Dr. Tony Xu Han, Assistant Professor, Dept. of Electrical and Computer Engineering

---

Dr. Ye Duan, Associate Professor, Dept. of Computer Science

Acknowledgements

I would like to thank Professor DeSouza whose assistance and guidance made the completion of this thesis possible.

Abstract

In this work the task of planar detection in an image pair is cast as an incomplete data problem where the parameters to be estimated are the ones that define the homographies induced by the planar regions in the scene. This incomplete data problem motivates the employment of the Expectation Maximization (EM) algorithm. Derivation of the EM algorithm equations proves that a closed form solution to the maximization step is impractical which leads to the proposal of a Modified Expectation Maximization (MEM) algorithm. The MEM algorithm presented replaces the traditional maximization step with an optimization based maximization step or a Kalman Filter based maximization step. In addition to this, recommendations are provided to reduce the number of parameters that need to be estimated by the MEM algorithm depending on the constraints of the scene. Experimental results show that the proposed MEM algorithm achieves comparable results to current methods for planar detection.

Contents

Acknowledgements ii
Abstract iii
List of Figures vi
List of Tables viii

1 Introduction **1**
1.1 Problem Statement 1
1.2 Approach and Contributions 2

2 Related Work **4**
2.1 Single Plane Detection 5
2.2 Multiple Plane 10

3 Background **16**
3.1 Camera Model 16
3.2 Planar Homography Constraint 17
3.3 Estimation of Homography 20
3.3.1 Minimizing Algebraic Error 20
3.3.2 Approaches Using Geometric Error 21
3.4 Decomposition of Homography 22
3.5 Representation of Homography 22

4 Clustering Homographies Using EM **25**
4.1 Expectation Maximization 25
4.2 Data Model 26
4.3 Case 1: Clustering Homography Parameters 28

4.3.1 Inferring Homographies 31
4.4 Case 2: Clustering Points 32
4.4.1 Approximate Likelihood Function 33
4.4.2 Applying Expectation Maximization 35
4.4.3 Modified Expectation Maximization (MEM) 38
4.5 MEM using Optimization 38
4.6 MEM using Kalman Filtering 40
4.6.1 The Kalman Filter 40
4.6.2 Kalman Filter Based M-Step 42
4.7 Kalman Filter For Motion Parameters 45
4.7.1 3D Reconstruction Using Rays 48
4.7.2 Kalman Filter Derivation 52
4.8 Case 3: Single Plane Detection 53
4.9 Variations of MEM 55

5 Results **56**
5.1 Modeling the Error Covariance: Σ_x 56
5.2 Ground Plane Detection For Mobile Robot 58
5.2.1 Comparison With Other Planar Detection Algorithms 62
5.2.2 Initial Estimate Refinement 65
5.3 Single Plane Tracking 67
5.3.1 Importance of Initial Estimate 71
5.3.2 The parameter Σ_x and its Effects 76
5.4 Multiple Planes - Sensitivity Analysis 78
5.4.1 Comparison of Variations 81

6 Future Work **86**

7 Conclusion **88**
Bibliography 90

List of Figures

3.1 Mapping the image point **x** from the first image to its corresponding image point **x'** in the second image using the homography **H** induced by the planar region π. 18

4.1 Graphical representation of a set of N point correspondences with observed data points x_n, x'_n and latent variables z_n, h_n. 27

4.2 Simplified graphical representation of a set of N homography vectors inferred from the point correspondences. 29

5.1 Flow chart for tracking and following algorithm 59

5.2 Sample images from test sequences. Black squares are pixels classified as ground. Blue circles are pixels classified as non-ground. 62

5.3 Average ground plane correspondence error per image 65

5.4 Sample images from Sequence 3. 70

5.5 Example where dominant plane is not the ground plane 72

5.6 Classification results for RANSAC and JLinkage for ground plane detection 72

5.7 SubSequence of Sequence 2. Configuration where $\sigma_u = \sigma_v = \sqrt{2}$ . . . 74

5.8 SubSequence of Sequence 2. Configuration where $\sigma_x = \sigma_y = 4$ 75

5.9 Sensitivity analysis results for optimization based MEM that solves for only the normal vector parameters and distance parameters 80

5.10 Sensitivity analysis results optimization based MEM that solves for global motion parameters in addition to normal vector parameters and distance parameters 82

5.11 Sensitivity analysis results for Kalman Filter based MEM 83

5.12 Example Classifications of the MEM algorithm for a multiple plane setting 85

List of Tables

5.1 Statistics of the Classification 62
5.2 Precision and Recall values for various parameter settings 64
5.3 Statistics of the Classification Compared with RANSAC and JLinkage 69

Chapter 1

Introduction

1.1 Problem Statement

Planar detection is an important topic in computer vision and many approaches have been proposed in the past decades. This problem is particularly interesting due to the innumerous planar structures found in both indoor and outdoor environments. The goal of any planar detection algorithm is to classify regions of an image into groups that correspond to the underlying planar structures of the scene. This classification is usually done using one of two approaches. In the first approach, classification is performed on a per pixel basis where every pixel in the entire image is classified and assigned a group. An alternative approach is a sparse form of the first where the classification is applied to point correspondences that are collected between two images.

The ability to detect planar regions in an image has benefits in a wide variety of areas such as autonomous robotics, augmented reality, structure from motion, camera self-calibration, and assistive technology. In autonomous robotics, planar detection can be employed to detect the ground plane of the scene. Detection of the ground plane is important for autonomous mobile robots because the pixels that are not classified as belonging to the ground plane can be viewed as obstacles to be avoided, as a target that needs to be followed, or as landmarks in the scene that need to be detected. Detection of obstacles can also be applied to assitive technology, such as devices for the blind which warn the user of where objects reside

in the environment, or a smart wheelchair that needs to act like an autonomous mobile robot. Planar detection can also be applied to augmented reality, where the classified point correspondences can be reconstructed to determine their underlying planar structure's parameters. Knowledge of the parameters that define the planar structures in a scene can then be used to superimpose objects onto the image. Last, the knowledge of planes in the environment and the homographies they induce can be employed in reconstruction problems such as structure from motion and camera self-calibration. In addition to this, algorithms can use planar detection to avoid, rather than classify planar points. One such example is the 8-point algorithm used to determine the fundamental matrix between camera pairs. In this case, the algorithm can avoid ill-conditioned solutions by eliminating coplanar points detected ahead of time in the scene.

1.2 Approach and Contributions

In this work, the problem of planar detection is applied to a set of point correspondences collected from two images. In this case, the homographies induced by the planar regions in the scene can be used as a criterion for classification of the point correspondences. That is, point correspondences belonging to the same specific planar region should uphold the homography constraint induced by the homography of that plane. Unfortunately, the only data that can be regarded as available is the set of point correspondences. So, in order to determine whether a group of points are coplanar, we need their planar homography, which is only available after the points are found to be coplanar – a "$chicken - egg$" problem.

Due to the absence of the homographies necessary for classification, our work poses the problem of planar detection as one of incomplete data where the parameters to be estimated are the ones that define the homographies induced by the planar regions in the scene. This motivates the employment of the Expectation Maximization (EM) algorithm to estimate the unknown homography parameters. However, as we will demonstrate later, the direct employment of the EM algorithm would result in a Maximization Step (M-Step) whose closed-form solution would be

impractical to derive.

Therefore, the contributions of this research are as follows:

- a modified EM algorithm that allows for a practical solution to the M-Step using iterative optimization methods (e.g. Nelder-Mead Simplex and Levenberg–Marquardt)
- a modified EM algorithm where the M-Step incorporates Kalman Filtering
- an algorithm that employs Kalman Filtering to estimate the motion parameters that define the transformation observed between two images
- an exhaustive comparative analysis of our methods against various methods in the literature

Chapter 2

Related Work

The research area of planar detection has been in existence for a few decades, and many applications benefit from locating planar regions in images. During this time, a large number of algorithms have been proposed which can be applied to specific computer vision problems such as obstacle detecting, target tracking, structure from motion, and autocalibration to name a few. Even though the problems to which these algorithms are applied may be vastly different from one another, the algorithms themselves all have the common goal of locating regions of images that belong to the same planar regions in the scene. The purpose of this section is to provide an overview of some of the algorithms in the literature, specifically the ones which are comparable to the algorithms presented in this body of work.

Since a large number of algorithms have been developed over the years, the discussion of the algorithms cannot be done all at once. Instead, in this section, the algorithms are broken down into two categories: 1) Single Plane, which approach the problem as a single plane to detect (usually the dominant plane) and 2) Multiple Plane, which focus on the detection of all planes in the scene. Both categories are discussed in this section since the algorithms presented in this work can be applied to both categories of planar detection.

2.1 Single Plane Detection

The approaches that are applied to detecting a single plane in an image or set of images mainly focus on finding the dominant plane in the scene. Most of the approaches that will be mentioned in this section are applied to mobile robotics, where the dominant plane to be detected is the ground plane. This application of planar detection serves two purposes. First, detection of the ground plane is important because it is the area in the environment on which the mobile robot must move. Second, by determining the ground plane in a scene, objects can be segmented out of the images, which allows for many vision based tasks for mobile robots such as obstacle avoidance to be easily implemented. Mobile robotics, however, is not the only area that can benefit from detecting ground planes. Another example is the tracking application in [1] which uses ground plane detection as a preliminary step before tracking vehicles in the image sequence.

Approaches for ground plane detection vary greatly based on the features they use -e.g. color, texture, but also on the frameworks that they employ to solving the task. The approaches described here were chosen to highlight the variety of algorithms available today. While our list does not cover all of the approaches presented in the literature, our purpose is to simply illustrate the key areas and the variety of techniques that are applied to ground plane detection.

Some of the simplest approaches to ground plane detection focus on modeling physical properties of the ground plane such as color and texture. In [2], for instance, a vision system using a single camera is proposed to detect obstacles in a scene by observing occlusions of the ground plane. Segmentation of the ground plane in an image is done by first learning a multi-modal distribution that captures color information about the ground plane. By using a multi-modal distribution, ground planes which contain multiple colors can be learned and segmented from the image. While this approach offers advantages over methods that require multiple cameras, the approach requires that a new model be learned for each new environment that is encountered. Also, for relying solely on color information for segmentation,it requires that the color of the obstacles be different from that of the ground plane.

Other approaches incorporate the use of optical flow [3] to determine the ground

plane in images [4,5]. In [4], the authors assume that the camera displacement between two consecutive images is small enough so the homography of the dominant plane can be approximated by an affine transformation. In their work, the optical flow is computed between two images and this affine transformation is computed by selecting three random points in the images. Using this affine transformation, the authors estimate a planar flow field and match it to the computed optical flow done in the previous step. If the matched region occupies more than half of the image, then the authors assume that the dominant plane has been detected, otherwise the process is repeated by selecting a new random set of points. The authors in [5] also incorporate optical flow for the segmentation of images into a two layered representation: (dominant plane and "other"). At first, the approach computes the optical flow using a coarse-to-fine estimation which provides a dense set of image point correspondences. The images are then split into small regions using a splitting technique based on color homogeneity and plane normals are estimated for all regions. After splitting the image into regions, an iterative refinement process is applied and a seed region is selected and grown by merging neighboring regions whose estimated plane normals are close to that of the seed region. The image is then decomposed into a two layer representation. One of the limiting factors of [4] is that the authors assume that the dominant plane occupies the majority of the image, which may not always be true. Also, optical flow calculations can be challenging when large homogeneous areas are present in the images.

Many algorithms also rely on 3D information about the scene to classify areas of the images as belonging to the ground plane. This 3D information is usually obtained from disparity maps or by stereo reconstruction of the scene [6–9]. In [7] for example, a stereo vision system is proposed to detect the ground plane and locate obstacles in the vehicle's path. The authors detect the ground plane by first obtaining a disparity map of the scene which is constructed from the stereo vision setup. This disparity map is then transformed into what the authors call the "V-disparity" map. This "V-disparity" map proves to be very useful since the ground plane in the scene corresponds to a line in the "V-disparity" map, which the system detects using the Hough Transform. After detecting the line corresponding to the

ground plane in the "V-disparity" image, segmentation of the ground plane from the original images becomes a trivial task. The authors in [9] extend the idea in [7] in two ways. First, the authors observe that the work in [7] is very dependent on the existence of distinct features on the ground plane. Their reasoning was that the correlation-based stereo matching used is prone to errors when there are no features present. To remedy this problem, the authors propose widening the size of the correlation window horizontally. Second, the authors eliminate the Hough transform for detecting the ground plane line in the "V-disparity" image by proposing to use a predefined set of ground lines that can be determined based on the stereo vision setup. Detection of the ground line in the V-disparity image using their method is done by choosing the predefined line that matches the current V-disparity image.

The approach in [6] differs from the previous methods in the use of disparity map. In their work, the authors first define a mapping between the ground plane and its corresponding disparity plane. Then they use this model to robustly fit a disparity plane to the data provided in the disparity map. This is accomplished by incorporating the Iteratively Re-weighted Least-Squares method. Once the a disparity plane is estimated the mapping is used to directly estimate the parameters of the ground plane.

While these 3D based methods all achieve robust results, they all suffer from the common problem of stereo vision algorithms which is knowing the precise extrinsic parameters of the stereo setup. This is crucial to the success of the algorithms since the accuracy of the disparity maps depend on the correct calibration of these parameters. Such calibration can be unfavorable for many applications where there exists a possibility for the cameras to move with respect to each other (even if slightly).

The authors of [10] overcome the problems inherit to stereo vision systems by proposing an algorithm that generates a depth map from a single image. The authors note that the human vision system rarely has trouble inferring structure of a scene from a single image and that this ability can be attributed to not only prior knowledge of the environment but also from monocular cues. This concept of monocular cues includes the expected variation of the appearance of texture at different depths

and the knowledge of the sizes of objects in the scene. In their approach, the authors use texture energies, texture gradients and haze to extract depth information from an image. Obtaining this information requires that an RGB image be converted into the YCbCr format with a single intensity value for each pixel. This image is then convolved with 17 filters to produce a 34 dimensional feature vector associated to each pixel. Besides the output from each filter, the feature vector also contains the squared energy of each filter. Finally, each pixel is represented by an absolute feature vector containing the 34 dimensional feature vector for the pixel and the feature vectors for the pixel's four neighbors. This 170 dimensional absolute feature vector is accompanied by a relative feature vector which represents the relative depths between two pixels. The authors then use these feature vectors to estimate the depth for every pixel in the image using a supervised learning algorithm applied on a Gaussian Markov Random field.

In addition to generating a depth map, the authors also use an algorithm to segment the image into regions that contain similar textures. The assumption is that similar textured areas are more likely to be coplanar, so the output of this algorithm provides different planar regions in the scene. The authors then use the depth map to merge the different planar regions in the scene to detect the ground plane.

The next approaches discussed here differ from the previous methods in that they focus on the classification of point correspondences which are collected from two images of a scene. Classification of these point correspondences is done using a projective property that is commonly referred to as the planar homography. The planar homography relation is described in great detail in the following chapter, but here, it suffices to say that these algorithms pose the problem of planar detection as a classification of the correspondences based on the estimated homography. This classification is done based on whether a point correspondence satisfies the homography constraint or not. The difficulty associated with this form of planar detection comes from estimating the homography using the point correspondences since not all the collected point correspondences derive from the dominant plane, and therefore they cannot be used to estimate the homography of the dominant plane. The following

algorithms provide methods for estimating the dominant homography from a set of correspondences which can then be used for classification.

The approach in [11] is interesting in that it employs an iterative two-step algorithm to determine the homography of the dominant plane between two views. The goal of this iterative two step method is not to obtain just the estimation of the homography, but also the estimation of the correspondences of the points between the two images. The first of this two-step method is the local step, which estimates the correspondence of the points. The correspondence for each point is done by taking the point in the first image and projecting it into the second image using the current estimate of the homography. The distance is computed between the projected point and all the points in the second image, with the correspondence being assigned to the point with the smallest distance. Once this is done for all of the points in the first image, the global step estimates the homography that maximizes this correspondence using robust estimation. This ensures that outliers will not have a significant effect on the estimation of the homography.

While accurate results are reported in [11], the algorithm suffers from initialization problems and contains the additional overhead of solving the correspondence problem while solving for the homography. Other planar detection algorithms treat the correspondence problem as a separate problem that can be solved by matching detected feature points using a descriptor such as the SIFT descriptor in [12]. Solving the correspondence problem in this manner allows for algorithms to focus solely on the estimation of the homography.

An example of such an approach is the H-clustering algorithm proposed in [13]. This algorithm begins by initializing a set of outliers containing all of the detected correspondences between the two images. A small set of correspondences is then constructed by choosing a random point correspondence and its four closest neighbors from the set of outliers. These five point correspondences are used to estimate a homography. Point correspondences from the set of outliers are then added to this set if they satisfy the homography constraint within an arbitrary threshold. This larger set of correspondences is then used to estimate another homography and the process is repeated until the number of inliers converges. If the resulting set contains

more than five point correspondences, the set is regarded as a cluster, otherwise the five correspondences are put back into the outlier set. In either case, the algorithm will repeat the previous steps by choosing a new set of five point correspondences until there are less than five correspondences in the outlier set. At this point, the resulting clusters are merged together to form the resulting dominant homography.

Other methods using point correspondence classification focus on estimating the homography of the dominant plane using a robust estimation technique such as the Least Median of Squares Estimator in [14]. However, the most popular and most commonly used robust estimation technique is the RANSAC method in [15]. Examples of algorithms that use RANSAC or RANSAC-like algorithms for robust estimation of the homography of the dominant plane are found in [16–18]. The RANSAC method starts with an estimate from the smallest possible data set. This small data set is referred to as the minimal sample set (MSS) and for the case of homography estimation has cardinality of four, since four point correspondences are required to determine a homography. Once a homography has been estimated from the sample set, a consensus set (CS) is derived by applying the homography constraint to all the point correspondences between two images. Point correspondences are included in the CS based on some criterion as for example the geometric error from the homography constraint being less than a certain threshold. If the resulting CS is sufficiently large, a homography is estimated using all the points in the CS, otherwise a new random set of four point correspondences is chosen and the algorithm iterates. RANSAC provides satisfactory results when estimating a homography from a set of point correspondences, however one drawback of the algorithm is that many MSS's may have to be drawn before a set that contains only coplanar point correspondences is found.

2.2 Multiple Plane

In this section, algorithms are presented whose objective is to locate multiple planar regions in an image. The extension of multiple plane detection from single plane detection is not trivial. This means that many of the previously described algorithms

cannot be used for multiple plane detection. Many of the algorithms discussed in this section follow the general idea of fitting a set of models to a data set. For planar detection, this data set contains the point correspondences between two images, while the models are the homographies that correctly map points from one image onto their corresponding points in the second image. Many times an error, usually referred to as the geometric error, is calculated to express how well a homography projects a point from the first image onto its corresponding position in the second image.

The approach in [19] accomplishes planar detection by employing projective invariants which are true if a set of five points are coplanar. This projective invariant is so called because it does not change under a projective transform. According to [19], the invariant calculated from five coplanar points in one image should be equivalent to the invariant calculated using their corresponding points in the second image. However, due to noise in the image this constraint can never be upheld, so the authors defined a way of thresholding the differences in the values of the invariants. Planar detection is done in [19] by first splitting up the point correspondences in the images into groups, each containing five correspondences. Next, a coplanarity test using invariants is applied to all groups to see if they are coplanar. If a group of points is determined to be coplanar, a homography is calculated using the point correspondences. This homography is then used to find other correspondences that may also belong to the group, and the homography is then re-estimated using the new set of correspondences. This framework allows for multiple planes to be detected from a scene.

In [20], a projective transformation called the planar affine homography is defined to determine planar regions between an image pair. Using a set of matched points between two images, a Delaunay triangulation is performed on the points in the first image, with a filtering scheme being used to discard triangles that are deemed to be degenerate. The matched points and the set of filtered triangles are then applied to a clustering procedure to determine which points are coplanar. The clustering procedure starts by taking a triangle from the set and using its points to calculate an affine homography that is added to an initially empty homography set. If the

geometric error of these correspondences using the affine homography is lower than a certain threshold, then the points are marked as belonging to that homography. All remaining triangles are then processed by first checking if any of the points that make up the triangle belong to an already defined homography in the homography set. If none of the points belong to a homography, then a new homography is calculated and added to the homography set. This procedure continues until all points are assigned to a homography. At the end of the clustering, only homographies which contain a number of points above a certain threshold are considered for point classification.

The Delaunay triangulation is also employed in the approach proposed in [21] where planes are detected from a set of images obtained using a stereo vision setup. Once again, the Delaunay triangulation is used to group points in the first image. After this grouping has occurred, a clustering technique similar to that in [20] is used to cluster the triangles. This process starts by selecting a triangle and calculating its normal using the author's proposed method which relies on the setup of the cameras. The triangle is then marked as visited and is placed in an initially empty triangle set noted as G_m. The process then goes through all of the triangles, starting with those adjacent to the first triangle and computes their normal vectors. If the normal vectors of neighboring triangles are similar to the first triangle's normal, they are placed into the same set of the first triangle. After all of the triangles are visited, triangles in the set G_m are removed from the set of all triangles and the process is repeated until no triangles are left to be visited. At the end of the process, several groups of coplanar triangles may be present. These groups are then sorted based on the number of elements they contain. The main planes are defined as the groups with the largest number of elements. Outlier points can also be detected since they are usually groups with only a single point. If the number of planes in the scene is known a-priori, then the authors state that the normal for each triangle can be computed, and a clustering algorithm such as k-means can be used to classify the triangles. While the authors propose a creative way to calculate the normal vector for each triangle, their method assumes that the transformations between the cameras in the stereo vision setup contain no rotation, strongly constraining this approach to a small number of applications.

Other methods of planar detection employ an extended version of RANSAC algorithm that can be used to detect multiple models. The first way to extend RANSAC in order to detect multiple models is through sequential RANSAC. This method sequentially applies RANSAC and removes inliers after each model is found. While this application of RANSAC makes logical sense, many approaches that use it for detecting multiple planar regions require further modifications of the algorithm.

For instance, [22, 23] modify the step in RANSAC that generates the minimal sample set (MSS). As the readers recall, the hypothesis is formed by randomly sampling a set of four point correspondences to estimate the model. When using this step in a multi-planar case the chances of choosing four points that are coplanar are significantly reduced. In [22] this step is modified by first selecting a single point at random, and then selecting the other three based on a conditional distribution that gives points closer to the selected point a higher chance of being selected. This modification also relies on the assumption that points closer to each other have a higher probability of being coplanar and this probability decreases as the set of points becomes spread across the image.

In [23], degenerate point sets are screened out based on how close they are to one another and if the points are collinear. The authors also mention a second strategy to choose four points that lay on two different edges of the image. Also, the authors use the determinant of the calculated homography to tell if the configuration is degenerate. Other than using these alternative methods to select the MSS's, the approaches in [22, 23] follow the basic sequential RANSAC method.

The authors of [24] argue that the use of sequential RANSAC is non optimal in that the incorrect removal of inliers from the data set can severely effect the estimates of the remaining models. To remedy this problem, the authors propose a parallel version of RANSAC that simultaneously estimates the parameters of all of the models which they call multi-RANSAC. Each iteration of the multi-RANSAC algorithm consists of drawing a MSS using a method similar to [22] and calculating the corresponding consensus set (CS). This is done for each model, with the inliers of the CS's being removed after each model is found. After the CS's are found for the iteration, the multi-RANSAC algorithm runs what the authors call a "Fusion

Procedure", where they combine the current CS's with the CS's found in the previous iteration. Once the CS's are updated, the algorithm updates the iteration threshold which is derived by the authors.

Multi-RANSAC proves to be an acceptable method when the models do not intersect each other. However, it suffers from the constraint that the number of models must be specified a-priori. This problem is overcome by the J-Linkage algorithm proposed in [25].

J-Linkage is another variant of the RANSAC algorithm which can be used to detect multiple models from data. In J-Linkage, M MSS's are drawn at the same time to estimate M models. These sets are drawn in a similar manner to [22, 24] where neighboring points are selected with a higher probability. Consensus sets for each model are then computed. It is noted that this differs from multi-RANSAC in that inliers are never removed from the data set. Each consensus set can then be used to define a preference set (PS) for each point, that is, each point has a set of estimated models it belongs to. The PS is used as a conceptual representation of the point, with the assumption that points which belong to the same structure will have the same PS's. Agglomerative clustering is then used to cluster the points based on their PS's. The distance metric used in this clustering algorithm is the Jaccard distance which gives the algorithm its name: J-Linkage.

J-Linkage has been shown to be able to fit many different types of models from lines to planes. However, one of the limiting factors of J-Linkage is the extremely large number of MSS's that need to be sampled to achieve a correct classification.

The last approach presented in this section is different from the rest in that it provides an algebraic solution to the problem of planar segmentation. The authors of [26] present two methods of planar segmentation that they say can be used as an initialization to other algorithms. The first approach presented in [26] embeds the homography relation into a system of quadratic polynomials. While this approach yields better classification results then the other algorithm presented, the dimension of the embedding and complexity are very large and may exceed the computer hardware limit when more than four planes need to be segmented.

In addition to this approach, the authors present a method that embeds the

homography constraint using a complex linear embedding. This embedding allows for a complex homography constraint to be defined that is equivalent to zero. This is beneficial, because it allows a system of equations to be defined by stacking the homography constraints of all of the point correspondences. The solution to this system of equations can be found using the singular value decomposition or the Rayleigh Quotient. and is called the multibody homography. Once the multibody homography has been obtained, the authors calculate a complex epipolar line for each point correspondence. The key is that the complex epipolar lines of coplanar point correspondences all intersect each other at a single point which is called the complex epipole. This provides a way of clustering the point correspondences based on finding the complex epipoles.

The approaches in [26] are interesting in that closed form solutions are presented to the planar segmentation problem. However, these approaches require that all point correspondences belong to a planar region in the scene. The presence of outliers in the set of point correspondences will affect the calculation of the multibody homography which will affect the calculation of the complex epipolar lines. A possible way to account for outliers would be to use an embedding that takes into account an large number of planes which allows for the outliers to belong to their own planes. However, this embedding has the tendency to become extremely large as the number of outliers increases.

Chapter 3

Background

In this section, we present an overview of the many concepts used by the algorithms proposed in this work. Here, we also establish the notation used throughout this thesis.

3.1 Camera Model

When a camera is used to capture an image of a scene, the 3D scene is being projected onto the 2D image plane of the camera. This mapping between 3D points in space and their corresponding 2D pixel coordinates (positions on the image plane) is usually specified by a matrix which represents a camera model. Of all of the camera models that have been developed, the simplest and most used is the pinhole camera model described in [17]. In order to represent the camera model as a linear mapping, homogeneous coordinates are used to represent points in both spaces, with a 3D point being denoted as $P = [X, Y, Z, 1]^T$ and a pixel being denoted by $p = [u, v, 1]^T$. The matrix that defines the camera model is called the camera calibration matrix shown in (3.1):

$$A = \begin{bmatrix} \alpha_u & s & c_u \\ 0 & \alpha_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

where α_u and α_v represent the focal length in both the u and v direction in

terms of pixel dimensions, s is the skew parameter which is zero for most cameras and c_u and c_v are the pixel coordinates of the projection of the camera center on the image plane. All of the parameters shown in (3.1) can be determined using a calibration technique such as the one in [27]. Using this model, the pixel coordinate of a 3D point may be calculated by multiplying the point by the camera calibration matrix. The normalization of this pixel coordinate is done with the z coordinate of the 3D point since it is assumed that the coordinate frame attached to the camera has its z-axis identical to the camera's optical axis. However, a consequence of this normalization is that all depth information is lost when projecting a 3D scene onto a 2D image.

3.2 Planar Homography Constraint

When a plane is imaged from two different perspective views, a homography between the two images is said to be induced by the plane. In projective geometry a homography is a projective transformation – an invertible 3x3 matrix – that maps points from one projective space to another. This transformation takes the form $\mathbf{x}' = H\mathbf{x}$ which is shown in more detail below:

$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ w \end{pmatrix} \tag{3.2}$$

Since points on the two image planes are represented by homogeneous coordinates, the image planes can be treated as a projective spaces which uphold this mapping. A homography that is induced by a plane can therefore be used to map points in one image to their corresponding positions in the image view. Figure 3.1 provides an example of this mapping.

Normally, however, the image planes are treated as euclidean coordinate systems and the homography mapping is represented as $\mathbf{x}' \sim H\mathbf{x}$ meaning that the product Hx is equal to $\mathbf{x}'$ up to a scale factor, which may require renormalization by the third coordinate to be equivalent to the homogeneous point $\mathbf{x}'$. This means that H

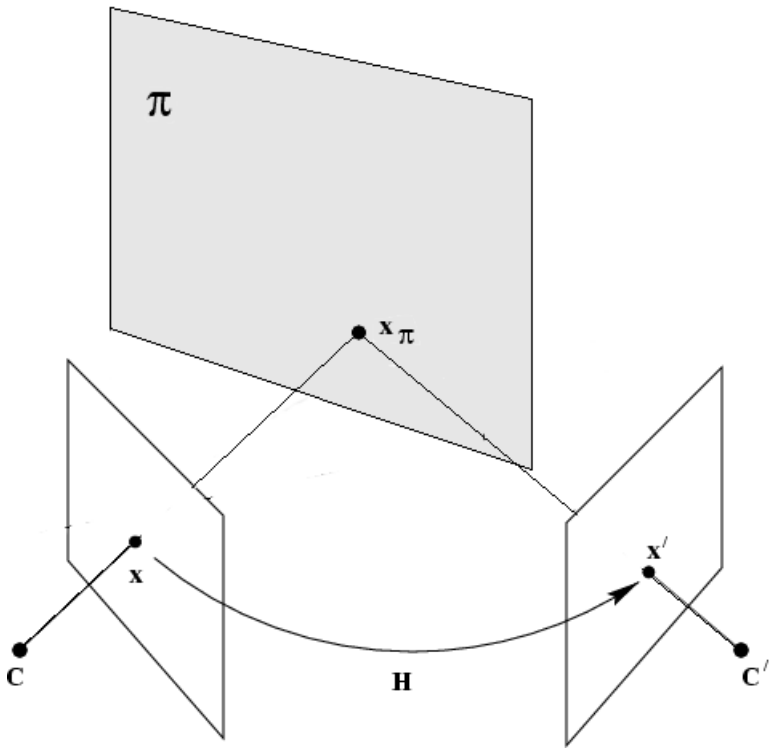


Figure 3.1: Mapping the image point **x** from the first image to its corresponding image point **x'** in the second image using the homography **H** induced by the planar region π.

can be multiplied by any non-zero scale factor without changing the mapping. Due to this property, H is called a homogeneous matrix and contains only eight degrees of freedom despite having nine elements.

Since this mapping is induced by the plane in the scene, it is only true for points in the image that belong to the plane. That is, points in the images that are not on the plane will not satisfy this property. In this work, the homography induced by the plane will be used to determine how likely corresponding points between two images belong to a planar region. Using this idea, the mapping in (3.2) can be rearranged to obtain two different types of error: algebraic and geometric.

The algebraic error distance is obtained by taking the norm of the cross product of the points $\mathbf{x}'$ and $H\mathbf{x}$. This is done by representing $\mathbf{x}'$ as a skew symmetric matrix. The algebraic error is shown below:

$$err_{alg} = \begin{bmatrix} 0 & -w' & y' \\ w' & 0 & -x' \\ -y' & x' & 0 \end{bmatrix} H\mathbf{x} \tag{3.3}$$

This equation provides a 3x1 error vector that must be normalized to obtain the algebraic error distance. While there exist three elements in this error vector coming from the three equations, [17] states that only two of the equations are linearly independent. This leads to the algebraic error distance being calculated using the norm of the first two elements in err_{alg}. One property to note about the algebraic error distance is that its value does not have any geometric or statistical meaning.

Geometric error, however, does have a statistical meaning. Computing geometric error is relatively simple in that it is the euclidean distance between two inhomogeneous points on the image plane. This requires that the two points $\mathbf{x}'$ and $H\mathbf{x}$ must be divided by their third term to obtain their inhomogeneous point representations on the image plane.

3.3 Estimation of Homography

The previous section presented the concept of the homography and two different error values that can be calculated using a homography. The purpose of this section is to describe some methods that use these error values to estimate the homography matrix when provided a set of point correspondences. These estimation techniques will be referenced when the algorithms in this work are presented. For a more in depth description of homography estimation techniques, the reader is referred to [28].

Before estimating a homography it is first important to determine the minimal amount of point correspondences that are needed to fully determine a homography. By inspecting the two types of error in the previous section, one can notice that a single point correspondence provides two constraints on the homography matrix. These two degrees of freedom correspond to a constraint in the u direction of the image plane and one in the v direction of the image plane. Knowing that each point correspondence provides two constraints and that the homography matrix has eight degrees of freedom, it is easy to determine that four point correspondences are needed in order to fully constrain the homography.

3.3.1 Minimizing Algebraic Error

The first estimation method presented is the Direct Linear Transformation algorithm (DLT). This algorithm minimizes the algebraic error distance in Section 3.2, because it allows for the homography to be determined using a simple linear solution. The algebraic error in (3.3) can be rewritten as a linear equation on the homography:

$$\begin{bmatrix} 0^T & -w'_i\mathbf{x_i}^T & y'_i\mathbf{x_i}^T \\ w'_i\mathbf{x_i}^T & 0^T & -x'_i\mathbf{x_i}^T \\ -y'_i\mathbf{x_i}^T & x'_i\mathbf{x_i^T} & 0^T \end{bmatrix} \mathbf{h} = A_i\mathbf{h} = 0 \tag{3.4}$$

where the vector $\mathbf{h} = [h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33}]^T$ is the stacked version of the homography matrix and A_i is a 3x9 matrix composed of the i^{th} point correspondence's values. As mentioned previously, while the algebraic error has three equations on $\mathbf{h}$, only two of them are linearly independent and it is common

to only use the first two rows of A_i.

To determine a homography from four point correspondences, the corresponding A_i matrices can be stacked to form a set of equations $A\mathbf{h} = 0$. Assuming that three of the point correspondences are not collinear, A is a matrix with rank 8 which has a 1 dimensional null space where the solution for $\mathbf{h}$ resides. When more than four point correspondences are provided, the solution is said to be overdetermined and can be solved using the singular value decomposition of A, where $\mathbf{h}$ is the singular vector associated with the smallest singular value of A.

The DLT algorithm is commonly used to estimate homographies due to its linear solution which results in a less computationally expensive algorithm. However, one disadvantage of this algorithm is that it minimizes the algebraic error, which has no geometric meaning. Despite this fact, the DLT algorithm can be successful when the data is properly normalized.

3.3.2 Approaches Using Geometric Error

Another set of homography estimation techniques focus on minimizing some form of geometric error described in Section 3.2. These estimation techniques usually rely on a non-linear optimization to determine the best values in $\mathbf{h}$ that minimizes the sum of geometric errors. Unfortunately, some optimization techniques require an initial estimate of the values of $\mathbf{h}$. Depending on the optimization, this estimate may or may not have a crucial impact on the final values of $\mathbf{h}$. Some instances may even use the algebraic solution previously described to obtain an initial estimate, and refine this estimate using these optimizations.

Other methods assume that the observed geometric error comes from noise and without this noise the geometric error would be zero. These methods take a statistical approach and assume that the noise can be modeled by a Gaussian probability distribution. The probability distribution of a point correspondence is then defined by:

$$P(\mathbf{x}'_i|H) = \frac{1}{2\pi\sigma^2} exp(\frac{-d(\mathbf{x}'_i, H\mathbf{x}_i)^2}{2\sigma^2}) \tag{3.5}$$

where $d(\mathbf{x}'_i, H\mathbf{x}_i)$ is a function that computes the geometric distance between the two points.

The log likelihood of the entire set of points then becomes:

$$log\ P(\mathbf{x}'_i|H) = -\frac{1}{2\sigma^2}\sum d(\mathbf{x}'_i, H\mathbf{x}_i)^2 + constant \tag{3.6}$$

The value of H that maximizes this likelihood is the one that minimizes the sum of squared geometric errors of all the point correspondences. This estimation of H is therefore equivalent to the estimate obtained by the approaches that use a non-linear optimization.

3.4 Decomposition of Homography

In many applications it may be necessary to decompose the homography into the following components:

$$H = A_2(R + \frac{t}{d}n^T)A_1^{-1} \tag{3.7}$$

where $A_1$and A_2 are the camera calibration matrices for each image, R and t are the rotation matrix and translation vector that form the transformation between the coordinate frames of the two cameras that captured the images, n is the normal vector of the plane, and d is the distance between the origin of the coordinate frame for image one and the plane.

Derivation of the decomposition of a homography into its corresponding R, t, n, and d is omitted for brevity. Instead, readers are referred to [29] where a derivation of the homography decomposition is provided in full detail.

3.5 Representation of Homography

The decomposition provided in the previous section explained how a homography can be decomposed into a rotation matrix, translation vector, normal vector, and distance scalar. These four entities contain 16 values i.e. nine values in the rotation

matrix, three values for each vector and one value for the distance scalar. However, in Section 3.2 it was stated that the homography is a homogeneous matrix which contains eight degrees of freedom. The purpose of this section is to describe how the homography matrix is defined in this work using only eight parameters.

To begin, it is noted that not all 3x3 matrices are valid rotation matrices. This places a constraint on the values of the nine parameters in R. Representation of a rotation matrix can be done using a standard representation such as Euler angles, the Rodrigues formula, or roll-pitch-yaw (RPY), all of which only require three parameters. In this work, R will be defined using the RPY formulation. Using the RPY formulation, a rotation matrix is constructed by the multiplication of three separate rotation matrices: a rotation of r around the x-axis, a rotation of p around the y-axis, and a rotation of y around the z-axis. The resulting rotation matrix is shown below:

$$R = \begin{bmatrix} cos(y)cos(p) & -sin(y)cos(r)+cos(y)sin(p)sin(r) & sin(y)sin(r)+cos(y)sin(p)cos(r) \\ sin(y)cos(p) & cos(y)cos(r)+sin(y)sin(p)sin(r) & -cos(y)sin(r)+sin(y)sin(p)cos(r) \\ -sin(p) & cos(p)sin(r) & cos(p)cos(r) \end{bmatrix} \tag{3.8}$$

Next, it is noted that the normal vector that contains three values also has a constraint which is that the norm of the vector must be unity. Using this constraint allows for the normal vector to be described using spherical coordinates. One will recall that spherical coordinates are defined using three values, the radial distance from the origin to the point, the colatitude angle φ, and the azimuth angle ϕ. However, the constraint of the norm of the normal vector being unity means that the radius value of the spherical coordinates will always be unity, which results in the normal vector being described by two angles which will be noted as φ_n andϕ_n. A normal vector is constructed using these angles in the following manner:

$$n = \begin{bmatrix} cos(\phi_n)sin(\varphi_n) \\ sin(\phi_n)sin(\varphi_n) \\ cos(\varphi_n) \end{bmatrix} \tag{3.9}$$

At this point, R and n are defined by five parameters which leaves three parameters to define the translation vector t and the distance value d. One representation could combine t and d together to create a vector $\frac{t}{d}$ which has three degrees of freedom. This, however has the potential to cause problems because t and d will be related to each other. Instead this work represents t and d independently where, t is represented as a unit vector, in the same manner as n, with angles φ_t and ϕ_t and d is its own degree of freedom.

$$t = \begin{bmatrix} cos(\phi_t)sin(\varphi_t) \\ sin(\phi_t)sin(\varphi_t) \\ cos(\varphi_t) \end{bmatrix} \tag{3.10}$$

Using the representations described, a homography is defined by the following parameter vector:

$$\theta = [r, p, y, \varphi_t, \phi_t, \varphi_n, \phi_n, d]^T \tag{3.11}$$

It is noted that this representation does contain singularities since spherical coordinates are used to define the normal and translation vectors.

The three angles that make up the rotation matrix (r, p, y) and the two angles that define the translation vector (φ_t, ϕ_t) will be referred to as the motion parameters in this work. The remaining parameters in (3.11) will be referred to as the plane parameters since these are the parameters which define a unique planar homography. Treating this set of parameters as two subsets of parameters does have advantages which will be discussed in the next section.

Chapter 4

Clustering Homographies Using EM

As stated in Section 1.2, this work proposes a set of algorithms that can be used to segment planar regions in two images of a scene. These algorithms treat the problem of planar segmentation as an incomplete data problem where the parameters to be estimated are the ones that define the homographies induced by the planar regions in the scene. This incomplete data problem motivates the employment of the Expectation Maximization (EM) algorithm in [30] along with a likelihood function that is based upon the homography constraint described in Section 3.2. In this section, a brief overview of the Expectation Maximization algorithm is provided followed by a model that can be used to describe how the point correspondences are observed. This section then presents the proposed algorithms for planar segmentation that use EM to cluster the point correspondences.

4.1 Expectation Maximization

First generalized by Dempster et. al in [30], the Expectation Maximization (EM) algorithm is a method commonly used for calculating maximum likelihood estimates of parameters of statistical models from incomplete data. The problem of incomplete data can take on many forms ranging from observation data that is missing to latent variables that cannot be directly observed. In these cases, there exists a likelihood function that depends on the **complete data set** which is comprised of both the observed data and the missing data. In order to estimate the model parameters

which maximize this likelihood, the EM algorithm iterates between an expectation step and a maximization step. In the expectation step, or E-Step, the expectation of the likelihood function with respect to the missing data is calculated based on the current estimate of the model parameters. In the maximization step or M-Step, a new set of model parameters is calculated which maximizes the expected likelihood derived in the E-Step. The algorithm iterates between these two steps until a convergence criterion is met. While this algorithm has a wide applicability to many problems, this section focuses on the use of EM for unsupervised clustering.

The application of EM for clustering is common in pattern recognition. In a typical clustering application, observed data needs to be grouped into different clusters based on models whose parameters are unknown. In this application, the latent variable is the label that associates each observation to the model that generated it. The combination of the observed data and the unobserved labels form the complete data set. In the E-Step, the expectation of the complete data likelihood is taken with respect to the labels, while the M-Step calculates a new set of parameters that maximizes this likelihood. When the algorithm converges, the model parameters are known and clustering can be done based on which model best describes the observation.

4.2 Data Model

Before describing how EM is applied to the problem of planar segmentation it is important to establish a model that illustrates the underlying process that produces the set of point correspondences observed between two images. This model will help to identify the variables in the complete data set and the model parameters that the EM algorithm will need to estimate. Figure 4.1 provides a graphical representation of this underlying process.

By looking at the model, it can be seen that four nodes exist which correspond to the four variables that make up the complete data set. The first node labeled x_n represents the variable which corresponds to the point detected in the first image having coordinates $[u, v]^T$. This point is assumed to be drawn from a discrete

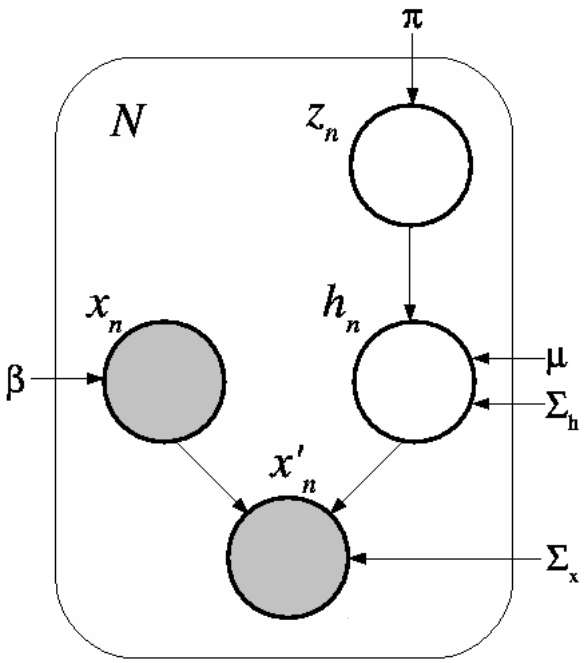


Figure 4.1: Graphical representation of a set of N point correspondences with observed data points x_n, x'_n and latent variables z_n, h_n.

uniform random distribution whose domain is the image plane. The second node denoted by z_n represents the label of the data and is drawn from a discrete uniform random distribution defined by $\pi = [\pi_1 ... \pi_k ... \pi_K]^T$ which contains the probability of observing a label z_n. This label corresponds to the k^{th} planar region that the observed point correspondence will belong to. The third node, denoted by h_n, is an eight dimensional vector of homography parameters shown in equation (3.11). This variable is observed from a Gaussian mixture model defined by the set of mean vectors in μ and set of covariance matrices in Σ_h. Each of the K Gaussian distributions corresponds to one of the K planar regions in the scene. That is, the mean vector of each of these Gaussians is a vector μ_k which contains the parameters of the homography that is induced by the k^{th} plane in the scene. The arrow between the nodes of z_n and h_n indicates that the Gaussian which h_n is observed from depends on the label z_n.

The last node in the graph labeled x'_n represents the variable that corresponds to the point observed in the second image with coordinates $[u', v']^T$. As indicated by the arrows in the figure, x'_n is dependent on the observed values of x_n and h_n.

Specifically, x'_n is defined by a Gaussian distribution with a mean that is defined by projecting the observed value of x_n into the second image using the homography transformation that is defined by the parameters in h_n. This mean value solidifies the fact that the point correspondence x_n and x'_n belongs to the planar region associated with label z_n and will uphold the homography constraint since x'_n was generated using x_n and h_n. We note that in order to use the homography constraint, the points x_n and x'_n must be represented using homogeneous coordinates. This is done by simply appending a one to each of the points coordinates to give the homogeneous coordinates $x_n = [u, v, 1]^T$ and $x'_n = [u', v', 1]^T$. The covariance Σ_x of this distribution is used to model the noise that occurs from matching corresponding points in two images.

4.3 Case 1: Clustering Homography Parameters

Using the model previously defined we now present our first approach that uses the EM algorithm to classify point correspondences. From the model we note that the only observed data is the point correspondences between the two images, while homography $\mathbf{h}_n$ and label z_n are latent variables. We immediately notice that this problem can be simplified if we could infer the latent homographies using the point correspondences. This would reduce the graph in Figure 4.1 to the one in Figure 4.2.

In Figure 4.2 the observed points have been eliminated and are represented by an observed homography which is inferred from the point correspondences. Calculating a homography for each point correspondence is not a trivial task and is discussed in detail in Section 4.3.1. The rest of this section assumes that a homography has been assigned to each correspondence using one of the methods described in that section.

Assuming that a homography can be calculated for each point correspondence, classification can be done based on which Gaussian in the mixture model best describes the observed homography. Unfortunately, the model parameters μ and Σ_h of these Gaussians are unknown and need to be estimated from the observed homogra-

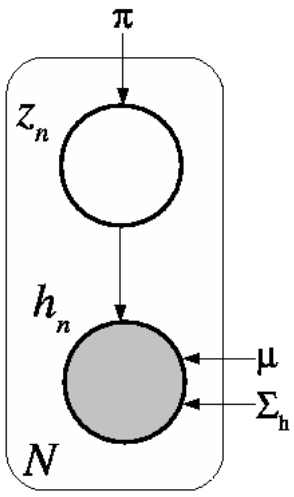


Figure 4.2: Simplified graphical representation of a set of N homography vectors inferred from the point correspondences.

phies. However, this estimation cannot be done because the labels of the observed homographies are unknown.

We recall, that the purpose of the EM algorithm is to solve the exact problem we just described where model parameters need to be estimated from an incomplete data set. Application of the EM algorithm requires us to provide a likelihood of the complete data set which consists of the observed homographies and the unknown labels based on a set of model parameters. This likelihood is defined as the following:

$$p(\mathcal{H}, Z|\mu, \mathbf{\Sigma}_h, \pi) = \prod_{n=1}^{N}\prod_{k=1}^{K}\mathcal{N}(\mathbf{h}_n|\mu_k, \Sigma_{hk})^{z_{nk}}\pi_k^{z_{nk}}$$

where μ and Σ_h are the set of means and covariances of the Gaussians in the mixture model, $\mathcal{H}$ is the set of inferred homographies, and Z is the set of unobserved labels. In this equation, the value of z_{nk} corresponds to the label of the n^{th} observed homography. This variable takes a value of one when k is equal to the observation's correct label and zero otherwise. It is easier to work with the log likelihood of the complete data, which is provided below:

$$ln\, p(\mathcal{H}, Z|\mu, \mathbf{\Sigma}_h, \pi) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\{ln\, \pi_k + ln\, \mathcal{N}(\mathbf{h}_n|\mu_k, \Sigma_{hk})\} \tag{4.1}$$

The E-Step consists of taking the expectation of the log likelihood in (4.1) with

respect to the latent variables in Z which results in the following equation:

$$\mathbb{E}[ln\, p(\mathcal{H}, Z|\mu, \mathbf{\Sigma}_h, \pi)] = \sum_{n=1}^{N}\sum_{k=1}^{K} p(z_{nk}|\mathbf{h}_n)\{ln\, \pi_k + ln\, \mathcal{N}(\mathbf{h}_n|\mu_k, \Sigma_{hk})\} \tag{4.2}$$

where $p(z_{nk}|\mathbf{h}_n)$ is defined using Bayes Theorem:

$$p(z_{nk}|\mathbf{h}_n) = \frac{\pi_k \mathcal{N}(\mathbf{h}_n|\mu_k, \Sigma_{hk})}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{h}_n|\mu_j, \Sigma_{hj})} \tag{4.3}$$

The M-Step then calculates a set of model parameters that maximizes the expected value of the log likelihood calculated in (4.2). Solving for the model parameters involves taking the derivative of (4.2) with respect to μ_k, Σ_{hk}, and π_k, setting these derivatives to zero to solve for μ_k, Σ_{hk}, and π_k. Since this application of EM is the common Gaussian mixture decomposition, these derivations are omitted and only the calculations for μ_k, Σ_{hk}, and π_k are provided:

$$\mu_k = \sum_{n=1}^{N} \frac{p(z_{nk}|\mathbf{h}_n)\mathbf{h}_n}{p(z_{nk}|\mathbf{h}_n)} \tag{4.4}$$

$$\Sigma_{hk} = \sum_{n=1}^{N} \frac{p(z_{nk}|\mathbf{h}_n)(\mathbf{h}_n - \mu_k)(\mathbf{h}_n - \mu_k)^T}{p(z_{nk}|\mathbf{h}_n)} \tag{4.5}$$

$$\pi_k = \frac{\sum_{n=1}^{N} p(z_{nk}|\mathbf{h}_n)}{N} \tag{4.6}$$

Now that we have defined how EM can be used to estimate the model parameters, we can now present our algorithm for planar detection using EM. Our algorithm begins with the collection of point correspondences between two images. A homography is then calculated for each point correspondence using one of the methods in Section 4.3.1. We then use the EM algorithm to estimate the sets of means, covariances, and label priors in μ, Σ_h, and π using the calculated homographies. In the E-Step (4.3) is used to calculate the posterior probability of the labels, while equations (4.4), (4.5), and (4.6) use these probabilities to update the estimates of the model parameters in the M-Step. These steps iterate until the change in model pa-

rameters between iterations falls below a predefined threshold. After convergence, the model parameters are then used to classify each of the homographies. This classification can then be directly applied to the point correspondences.

4.3.1 Inferring Homographies

In the previous section, a classification algorithm was presented which makes a simplification to the model presented in Figure 4.1. This simplification requires that a homography be inferred for each point correspondence, which is not a trivial task since a homography has eight degrees of freedom while a point correspondence has two constraints. This requires that at least three other point correspondences must be used with the point correspondence that the homography is being calculated for. After a set of point correspondences has been found, the homography of the set of points can be determined using one of the methods in Section 3.3 and the individual homography parameters can be retrieved using the decomposition method mentioned in Section 3.4. The rest of this section describes some possible methods for obtaining a set of point correspondences that can be used to define a homography.

Before describing the methods for obtaining a set of correspondences we first provide two constraints that we impose on the set. The first constraint is that three or more points in the set cannot be collinear. Collinear points will not fully constrain the homography resulting in a degenerate homography calculation. Second, and most importantly, the points should be coplanar. Points that are not coplanar will provide a poor estimate of the homography since the homography that is fitted will try to minimize the mapping error for all of the correspondences in the set. A set of point correspondences that fits these constraints should provide a sufficient estimation of the homography parameters.

One of the first methods we present is the naive random sampling. While the collinearity constraint can be checked for each sampled point, there is no way to ensure that the points selected will be coplanar. A way to modify the random sampling is to adopt a method similar to the one in [22] where a prior distribution is set around the point correspondence we are constructing a set for. Even though this approach does not completely ensure that the set will be coplanar, it does promote

choosing points that are in close proximity to one another. These points will have a higher probability of belonging to the same planar region in the scene.

A second method that could be used is presented in [20]. In that work, point groups are selected based on a Delaunay triangulation. Groups can then be formed based on this triangulation. Similar to the previous method, this method promotes sets of points that are in close proximity to one another. However, one disadvantage of methods that promote close points is that points that are too close together run the risk of estimating a noisy homography, since the points contain noise from matching. The authors in [20] addressed this problem by adding a step that filtered triangles that did not cover a large enough area.

The last method we present uses the idea of the projective invariant mentioned in [19]. This invariant can be used to detect if a set of points are indeed coplanar. Using this coplanarity test, groups could be chosen in such a way to cover a certain sized area, while still upholding the coplanarity constraint.

While the previous methods of set selection can be used to estimate a homography for each point correspondence, we point out that each introduces some type of ad-hoc parameters that compromises the robustness of the methods. A better approach to planar detection would be to eliminate the need of estimating the homographies and applying the EM algorithm directly to the observed point correspondences.

4.4 Case 2: Clustering Points

The previous approach simplified the model in Figure 4.1 by attempting to assign a homography to each point correspondence. This allowed the problem of planar detection to be represented as a Gaussian mixture decomposition problem which used EM to obtain the model parameters. The simplification of the model, however, required that sets of points be grouped together in order to estimate a homography for each correspondence. By doing this, the robustness of the algorithm suffers since the point sets cannot completely guarantee that the estimated homography is a close approximate to the true homography that generated the data.

A more robust approach would be an algorithm that makes direct use of the observed point correspondences. However, directly using the point correspondences introduces a second latent variable which is the unobserved homography $\mathbf{h}_n$. This latent variable adds additional complexity to the EM algorithm, meaning that the expectation of the complete data log likelihood has to be taken with respect to the labels and the homographies. This motivates us to find a way to approximate the complete data likelihood so that its calculation does not depend on the observed homographies. By defining a likelihood that is independent of the homographies, the application of the EM will be significantly easier.

4.4.1 Approximate Likelihood Function

We begin by constructing the complete data likelihood from the graph in Figure 4.1 which takes the form:

$$p(X', X, \mathcal{H}, Z|\mu, \Sigma_h, \Sigma_x, \beta, \pi) = \prod_{n=1}^{N}\prod_{k=1}^{K} p(\mathbf{x}'_n, \mathbf{x}_n, \mathbf{h}_n|\mu_k, \Sigma_{hk}, \Sigma_x, \beta)^{z_{nk}} \pi_k^{z_{nk}} \tag{4.7}$$

where X and X' are the sets of observed point correspondences, $\mathcal{H}$ is the set of homographies, Z is the set of labels, and β represents the set of parameters that govern the uniform distribution that X is observed from. The joint probability on the right hand side of (4.7) can be broken down into the product of conditional probabilities shown below:

$$p(\mathbf{x}'_n, \mathbf{x}_n, \mathbf{h}_n|\mu_k, \Sigma_{hk}, \Sigma_x, \beta) = p(\mathbf{x}'_n, \mathbf{h}_n|\mathbf{x}_n, \Sigma_x, \mu_k, \Sigma_{hk})\, p(\mathbf{x}_n|\beta) \tag{4.8}$$

Using this representation we notice that we have a joint probability distribution for $\mathbf{x}'_n$ and $\mathbf{h}_n$. This joint distribution can be broken down further into the following conditional probabilities:

$$p(\mathbf{x}'_n, \mathbf{h}_n|\mathbf{x}_n, \Sigma_x, \mu_k, \Sigma_{hk}) = p(\mathbf{x}'_n|\mathbf{h}_n, \mathbf{x}_n, \Sigma_x)\, p(\mathbf{h}_n|\mu_k, \Sigma_{hk}) \tag{4.9}$$

where we recall $p(\mathbf{x}'_n|\mathbf{h}_n, \mathbf{x}_n, \Sigma_x)$ is defined by a Gaussian whose mean is defined

by $\mathbf{h}_n$ and $\mathbf{x}_n$ and covariance Σ_x, while $p(\mathbf{h}_n|\mu_k, \Sigma_{hk})$ is a Gaussian with mean μ_k and covariance Σ_{hk}.

As we mentioned, our goal is to define an approximation to the complete data likelihood that is independent of the latent homographies. We notice that in (4.9) we can eliminate the latent homographies by marginalizing (4.9) with respect to $\mathbf{h}_n$. This is done by integrating (4.9) over all possible values of $\mathbf{h}_n$ shown below:

$$p(\mathbf{x}'_n|\mathbf{x}_n, \Sigma_x, \mu_k, \Sigma_{hk}) = \int_{\mathbf{h}_n} p(\mathbf{x}'_n, \mathbf{h}_n|\mathbf{x}_n, \Sigma_x, \mu_k, \Sigma_{hk}) \tag{4.10}$$

The integral in (4.10) may appear daunting, but we note that a constraint can be applied to the model in Figure 4.1. That is, we can make the assumption that the covariances of the Gaussians in the mixture model that produces $\mathbf{h}_n$ are equivalent to each other and are composed of extremely small variances. This allows us to take the limit of the integral in (4.10) which becomes:

$$\lim \int_{\mathbf{h}_n} p(\mathbf{x}'_n|\mathbf{h}_n, \mathbf{x}_n, \Sigma_x)\, p(\mathbf{h}_n|\mu_k, \Sigma_{hk}) = p(\mathbf{x}'_n|\mu_k, \mathbf{x}_n, \Sigma_x)$$

where $p(\mathbf{x}'_n|\mu_k, \mathbf{x}_n, \Sigma_x)$ is a Gaussian whose mean is defined by μ_k and $\mathbf{x}_n$.

Another way to visualize this approximation is to think of two one dimensional normally distributed variables W and Q where the mean of the distribution of W is Q. The joint distribution of W and Q is then written as

$$p(W, Q) = p(W|Q)p(Q) = \mathcal{N}(\,\mathcal{N}(\mu_Q, \sigma_Q)\,,\, \sigma_W)\,\mathcal{N}(\mu_Q, \sigma_Q)$$

where μ_Q, σ_Q , and σ_W are the mean of Q, variance of Q, and variance of W. Intuitively, one can see that as the variance of Q approaches zero, the joint distribution can be represented by $\mathcal{N}(\mu_Q\,,\, \sigma_W)$.

This assumption is valid to make with regards to planar detection because the underlying planes that generate the observed data should hardly contain any variance. In the previous approach, the covariances in Σ_h took into account the errors of inferring the homographies for each point correspondence. However, since this approach uses the point correspondences directly, the only error that is encountered comes from the matching algorithm that was used to obtain the point correspon-

dences which is described by Σ_x.

This approximation allows us to define the complete data likelihood as the following:

$$p(X', X, Z|\mu, \Sigma_h, \Sigma_x, \beta, \pi) = \prod_{n=1}^{N}\prod_{k=1}^{K}\{p(\mathbf{x}'_n|\mu_k, \Sigma_x, \mathbf{x}_n)\, p(\mathbf{x}_n|\beta)\}^{z_{nk}}\pi_k^{z_{nk}} \tag{4.11}$$

We note that in (4.11) this approximation of the complete data likelihood does not rely on the latent homographies.

4.4.2 Applying Expectation Maximization

In the previous section, we presented an approximation to the complete data likelihood. Using this likelihood we can now derive the equations used by the EM algorithm. We start by denoting which model parameters of Figure 4.1 are known and do not need to be estimated. First, our approximation in the previous section requires that we do not need to know the covariances in Σ_h since the assumption is that these variances are negligible. Also, we can assume that the parameters in β are known since we know the domain that the uniformly random distribution operates on. Lastly, the covariance Σ_x, which models the matching noise, can be learned empirically which we do in Section 5.1. This leaves us with the parameters in π and μ which need to be estimated. Once again, we encounter the problem where model parameters need to be estimated from an incomplete data set which we will solve by using the EM algorithm. Unfortunately, this application will not be as trivial as the previous approach, and will require us to modify the maximization step of the algorithm.

As before, the first step is the E-Step which requires us to take the expectation of the log of the likelihood function in (4.11) with respect to the latent variables of the model. Since our approximation to the complete data likelihood does not rely on the latent homographies, we only need to take the expectation with respect to the labels. The expectation of the log likelihood with respect to the latent variable z_{nk} is:

$$\mathbb{E}[ln\, p(X',X,Z|\mu,\Sigma_h,\Sigma_x,\beta,\pi)]=\sum_{n=1}^{N}\sum_{k=1}^{k} p(z_{nk}|\mathbf{x'}_n)\{ln\,\pi_k+ln\, p(\mathbf{x}'_n|\mu_k,\mathbf{x}_n,\Sigma_x)+ln\, p(\mathbf{x}_n|\beta)\} \tag{4.12}$$

where $p(z_{nk}|\mathbf{x'}_n)$ is defined using Bayes Theorem:

$$p(z_{nk}|\mathbf{x'}_n) = \frac{\pi_k\, p(\mathbf{x}'_n|\mu_k,\mathbf{x}_n,\Sigma_x)}{\sum_{j=1}^{K}\pi_j p(\mathbf{x}'_n|\mu_j,\mathbf{x}_n,\Sigma_x)} \tag{4.13}$$

We recall that the purpose of the following M-Step is to then calculate the values in π and μ that maximize the expected log likelihood in (4.12). The standard method to obtaining equations to calculate these maximum likelihood values requires us to set the partial derivatives of (4.12) with respect to π_k and μ_k to zero and solving for the values in π_k and μ_k. We use this method to derive the maximum likelihood update equation for π_k.

First, we note that there is a contraint on the values of π_k since they form a discrete distribution which is:

$$\sum_{k=1}^{K}\pi_k = 1$$

In order to take this constraint into account, a Lagrange multiplier is used when defining the expected log likelihood. We then take the derivative of the expected log likelihood and Lagrange multiplier with respect to π_k which is shown in the following:

$$\frac{\delta}{\delta\pi_k}\mathbb{E}[ln\, p(X',X,Z|\mu,\Sigma_h,\Sigma_x,\beta,\pi)] + \frac{\delta}{\delta\pi_k}\lambda\left(\sum_{k=1}^{K}\pi_k - 1\right) = \sum_{n=1}^{N} p(z_{nk}|\mathbf{x'}_n)\frac{1}{\pi_k} + \lambda \tag{4.14}$$

We then set the derivative in (4.14) to zero and rearrange the terms to get λ and π_k on the left hand side.

$$-\lambda\pi_k = \sum_{n=1}^{N} p(z_{nk}|\mathbf{x'}_n) \tag{4.15}$$

Solving for λ requires that both sides of (4.15) be summed over all values of k,

which results in $-\lambda = N$. Substituting $-N$ in (4.15) for λ and solving for π_k gives the following equation to calculate the maximum likelihood value of π_k.

$$\pi_k = \frac{\sum_{n=1}^{N} p(z_{nk}|\mathbf{x'}_n, \mathbf{x}_n)}{N} \tag{4.16}$$

Using the same technique to derive the maximum likelihood update equation for μ_k requires us to take the derivative of (4.12) with respect to μ_k and set it equal to zero:

$$\frac{\delta}{\delta \mu_k} \sum_{n=1}^{N} p(z_{nk}|\mathbf{x'}_n, \mathbf{x}_n) \left(-\frac{1}{2} f(\mathbf{x'}_n, \mu_k, \mathbf{x}_n)^T \Sigma_x^{-1} f(\mathbf{x'}_n, \mu_k, \mathbf{x}_n) \right) = 0 \tag{4.17}$$

In equation (4.17), we notice the introduction of the function $f(.)$. We recall that $p(\mathbf{x}'_n|\mu_k, \mathbf{x}_n, \Sigma_x)$ is a Gaussian with covariance Σ_x and a mean that is defined by μ_k and $\mathbf{x}_n$. The function $f(.)$ is used to calculate the difference between the mean of this distribution and the observed value of $\mathbf{x}'_n$. Specifically, this function takes the parameters in μ_k and calculates the corresponding rotation matrix, translation vector, normal vector, and distance value that define the homography induced by the planar region using the representations in (3.8), (3.10), and (3.9). These are then combined to form a homography using equation (3.7). The resulting homography H is used with $\mathbf{x}_n$ to find the mean value of the Gaussian distribution using the homography constraint. The function then calculates and returns the difference between $\mathbf{x}'_n$ and the mean value defined by H and $\mathbf{x}_n$ shown below:

$$f(\mathbf{x'}_n, \mu_k, \mathbf{x}_n) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$$

where

$$f_1 = \left(\frac{h_{11}u + h_{12}v + h_{13}}{h_{31}u + h_{32}v + h_{33}} \right) - u'$$

$$f_2 = \left(\frac{h_{21}u + h_{22}v + h_{23}}{h_{31}u + h_{32}v + h_{33}} \right) - v' \tag{4.18}$$

In (4.18) we notice the division by the third row of the homography parameters which converts the transformed point $H\mathbf{x}_n$ from the projective coordinate system to the euclidean coordinate system of the image plane.

Solving for the individual parameters in μ_k requires us to take the representations in (3.8), (3.10), and (3.9) to calculate the corresponding values of R, t, n, and d and factor them into the equation in (3.7). The resulting homography then needs to be factored into (4.18) which in turn needs to be substituted into (4.17). The equation in (4.17) then needs to be solved for the individual parameters in μ_k.

It is clear to see that this solution to calculating the μ_k that maximizes (4.12) is impractical. Instead, we propose two methods to avoid such a solution in the next two sections. Both provide a modified M-Step in our EM algorithm which allows for a maximum likelihood solution for μ_k to be calculated.

4.4.3 Modified Expectation Maximization (MEM)

Now that we have defined how EM can be used to estimate the model parameters, we can present our algorithm for planar detection using our modified EM algorithm. Our algorithm starts by collecting point correspondences between two images. We then use the modified EM algorithm to estimate the model parameters μ and π using the point correspondences. In the E-Step, (4.13) is used to calculate the posterior probability of the labels given the point correspondences. These posterior probabilities are then used in the modified M-Step which incorporates one of the following discussed methods to obtain the values of μ and π that maximize the expected log likelihood. These steps iterate until the change in model parameters between iterations falls below a threshold. After convergence, the model parameters are then used to classify each of the point correspondences based on their posterior probabilities.

4.5 MEM using Optimization

As previously mentioned, a closed form solution to determining the values in μ_k is impractical and a method is needed to determine the values in μ_k that maximize the

expected log likelihood in (4.12). We notice from (4.12) that the μ_k that maximizes the expected value of the log likelihood is the one that maximizes $p(\mathbf{x}'_n|\mu_k, \mathbf{x}_n, \Sigma_x)$ by minimizing the function $f(\mathbf{x}'_n, \mu_k, \mathbf{x}_n)$. Using this knowledge, one method for estimating μ_k is to employ a non-linear optimization whose objective function is based on $f(\mathbf{x}'_n, \mu_k, \mathbf{x}_n)$. The goal of this optimization is to minimize the sum of squared error values returned from $f(\mathbf{x}'_n, \mu_k, \mathbf{x}_n)$ for the set of point correspondences. This method is similar to the maximum likelihood solution for homography estimation provided in section 3.3.2.

The problem with this method is that the optimization will find a value for μ_k that minimizes the error from $f(\mathbf{x}'_n, \mu_k, \mathbf{x}_n)$ for all of the point correspondences in the image pair. Doing so will estimate a μ_k that will not describe the homography of the k^{th} planar region since the optimization uses all points correspondences which belong to several different planar regions. In order for this optimization to be successful, we need a way to minimize the effect of correspondences that do not belong to the k^{th} planar region. This is achieved by weighting the error values for a point correspondence by the correspondence's posterior probability $p(z_{nk}|\mathbf{x}'_n)$. This will weight the errors in such a way so that point correspondences with a higher probability of belonging to the k^{th} planar region are weighted heavier in the optimization while correspondences that do not belong to the k^{th} planar region will be weighted with low probabilities (ideally 0) and have little to no effect on the optimization. This weighted optimization reflects the original purpose of the M-Step which maximizes (4.12) while taking into account the posterior probabilities $p(z_{nk}|\mathbf{x}'_n)$.

For the implementations provided in this work, the optimization used is the Nelder-Mead Simplex algorithm [31]. This optimization was ideal because it uses only function evaluations to search. While this optimization was used exclusively in this work, we do note that any non-linear optimization should suffice for the modified M-Step. Our choice for the Nelder-Mead Simplex comes from its simple implementation and quick convergence.

4.6 MEM using Kalman Filtering

Using a non-linear optimization is an effective way to implement the modified M-Step of the MEM algorithm. In this section an alternative method based on Kalman Filtering is presented that can also be used as an M-Step for the MEM algorithm. This section begins by providing an overview of the Kalman Filter followed by the derivation of the Kalman Filter based M-Step. Following the derivation is a second application and derivation of Kalman filtering that can be applied to planar detection and can serve as a simplification to the MEM algorithm.

4.6.1 The Kalman Filter

The Kalman Filter [32] is a recursive filter that is used to estimate the state of a process from noisy measurements. The recursive nature of the Kalman Filter allows for the estimate of the state to be updated each time a new measurement is obtained without having to retain any of the previous measurements. Two equations are used to describe the process in question when using a Kalman Filter: 1) the equation that governs the process and 2) the measurement equation. The purpose of the first equation is to model the evolution of the state of the process over time. This equation takes the following form:

$$p_k = Ap_{k-1} + w_{k-1} \tag{4.19}$$

where p_k and p_{k-1} are the states of the system at times k and $k-1$ respectively, the matrix A models how the state changes over time, and w_{k-1} is the process noise that is assumed to follow a normal distribution with covariance Q. The measurement equation is used to relate the state to an observed measurement and takes the form:

$$y_k = Mp_k + u_k \tag{4.20}$$

where y_k is the observed measurement, M is the measurement matrix that relates the state to the measurement, p_k is the state of the system, and u_k is the observed measurement noise that follows a normal distribution with covariance U.

The Kalman Filter operates by iterating between two steps: Prediction and Update. In the prediction step, (4.19) is used to estimate the state of the system which will be denoted as p_k^-. This state prediction will have an uncertainty associated with it which is denoted as P_k^- and is calculated using the following equation:

$$P_k^- = AP_{k-1}A^T + Q$$

It is noted that both the estimated state and the uncertainty associated with the state contain super scripts of (-) to denote that these are the apriori estimates of these values, that is, the estimates of these values before a measurement is observed.

The update step of the Kalman Filter is used to determine the aposteriori values of p_k^- and P_k^- when a measurement is observed. This is done using the following equations:

$$p_k = p_k^- + K_k(y_k - Mp_k^-) \tag{4.21}$$

$$P_k = (1 - K_k M)P_k^- \tag{4.22}$$

Both (4.21) and (4.22) introduce the parameter, K_k, which is known as the Kalman Gain and is calculated using the equation below:

$$K_k = P_k^- M^T (MP_k^- M^T + U)^{-1} \tag{4.23}$$

The Kalman Gain can be viewed as a blending factor that combines the predicted value and the value of the state inferred from the noisy measurement to obtain an estimate that is more accurate than either one alone. Upon inspecting (4.23) it can be seen that when the uncertainty in the prediction (P_k^-) approaches zero, K_k goes to zero and the aposteriori estimate of the state is the apriori estimate. Conversely, if the uncertainty in the measurement (U) approaches zero, K_k goes to one and the aposteriori estimate of the state is updated based on the difference between the observed measurement and the predicted measurement.

4.6.2 Kalman Filter Based M-Step

After reviewing the Kalman Filter in Section 4.6.1 an explanation can now be provided on how the filter can be used to develop an alternative modified M-Step for the MEM algorithm. In order to apply Kalman Filtering in the M-Step we must first define the process whose state we are trying to estimate with the Kalman Filter. This process is the one illustrated in Figure 4.1 which produces the point correspondences observed in a planar detection problem. However when deriving the Kalman Filter based M-Step one simplification is required to be made to this process. Specifically, it will need to be assumed that only one label is generated in this process such that $\pi = 1$. From the figure it is easy to determine that the state of the process is contained in μ_k. One property to note is that the values in μ_k will not evolve over time. This simplifies (4.19) in the prediction step to:

$$p_k = p_{k-1} + w_{k-1} \tag{4.24}$$

where A has been replaced with the identity matrix. In the process in Figure 4.1 the parameters in μ_k are used to produce sets of point correspondences. These point correspondences will serve as the noisy measurements that the Kalman Filter will use to estimate the state.

This assignment of noisy measurements and state implies that the Kalman Filter based M-Step will be using the point correspondences to estimate the parameters in μ_k. This coincides perfectly with the purpose of the M-Step of the EM algorithm where the parameters in μ_k need to be updated using the point correspondences. Now that the state and measurements have been defined, we can proceed to derive the Kalman Filter equations needed in the Kalman Filter based M-Step.

Applying Kalman Filtering to the above process requires that any constraints on the state parameters must be linearized if they are non-linear. In our application we must first define the constraint equation. We note that (4.18) provides an equation that can be used as a constraint when set to 0. This equation is non-linear which means that it must be linearized. In order to linearize the equation we must take the partial derivatives with respect to the measured point correspondence and the

estimated values in μ_k. This results in the following equation:

$$f(\mathbf{x'}_n, \mu_k, \mathbf{x}_n) = f(\hat{x}'_n, \hat{\mu}_k, \mathbf{x}_n) + \frac{df}{dx_n}(\mathbf{x'}_n - \hat{x}'_n) + \frac{df}{d\mu_k}(\mu_k - \hat{\mu}_k) = 0 \tag{4.25}$$

In this equation, $\mathbf{x'}_n$ is the noiseless value of the point in the second image that makes up the correspondence and μ_k is the correct value of the parameters which we are trying to estimate. These are contrasted to $\hat{x}'_n$ and $\hat{\mu}_k$ which are the observed measurement and current estimate of the state vector. The above equation can then be rearranged to form the measurement equation in (4.20) that is necessary for the Kalman Filter where:

$$y_k = -f(\hat{x}'_n, \hat{\mu}_k, \mathbf{x}_n) + \frac{df}{d\mu_k}\hat{\mu}_k$$

$$M = \frac{df}{d\mu_k}$$

$$u_k = \frac{df}{dx_n}(\mathbf{x'}_n - \hat{x}'_n)$$

Another parameter of interest is the measurement error covariance U which is defined as:

$$U = \frac{df}{dx_n}\Sigma_x\frac{df}{dx_n}^T$$

where Σ_x is the error covariance from the matching algorithm in Figure 4.1.

Using this definition of the measurement equation, we can present the Kalman Filter based M-Step. In this M-Step, each point correspondence will be considered a noisy measurement that was obtained from the process. These noisy measurements will be used by the Kalman Filter to update the estimate of the parameters in μ_k. The prediction step of the Kalman Filter is simplified to the equation in (4.24) since the state vector does not evolve over time. In the update step of the Kalman Filter, the following equations are used to update the parameters in μ_k and their uncertainty:

$$\hat{\mu}_{k+1} = \hat{\mu}_k + K(y_k - M\hat{\mu}_k) \tag{4.26}$$

$$K = P_k M^T (U + M P_k M^T)^{-1}$$

$$P_{k+1} = (I - K)P_k$$

The Kalman Filter iterates until all of the point correspondences are used to update the parameters in μ_k. It is noted that in the Kalman Filter based M-Step this step is executed to estimate the values for each μ_k where $k = 1, .., k, .., K$ and K is the number of labels. Last we note that a similar problem exists for this M-Step that existed in the optimization based M-Step. This problem is that not all of the point correspondences are produced by the $\mu_k$that is being estimated. This would be the equivalent of using measurements from a different process to estimate the state of our process. To remedy this problem two steps can be taken. First, when we use the point correspondences to update μ_k we can enforce an order of which correspondences to use first. This order can be based on the posterior probabilities assigned to the point correspondences where the correspondences with the highest posterior probability are used first. The purpose of this strategy is based on the fact that the uncertainty in the state vector decreases with each sample that is used to estimate the state vector. This means that the Kalman gain (K) will decrease with each measurement that is used. By using the highest probability correspondences first, we allow these correspondences to have the greatest effect on the update of μ_k since the Kalman Gain will be larger for these correspondences. Another method that can be incorporated is to modify the update of μ_k in (4.26) to take into account the posterior probability of the correspondence as shown below:

$$\hat{\mu}_{k+1} = \hat{\mu}_k + p(z_{nk}|\mathbf{x'}_n)K(y_k - M\hat{\mu}_k)$$

This would ensure that correspondences that were not generated by μ_k do not contribute to its estimation since they will have low posterior probabilities.

In this section a modified M-Step that uses a Kalman Filter to estimate the parameters in $\mu_k$using the point correspondences was presented. This M-Step works by treating μ_k as the state vector and the point correspondences as noisy observations. Incorporating a Kalman Filter in the M-Step ensures that we are estimating the parameters in μ_k that generated the observed point correspondences, which is the original purpose of the M-Step of the EM algorithm. The next section will provide another application of the Kalman Filter that will reduce the number of parameters to be estimated by the MEM algorithm.

4.7 Kalman Filter For Motion Parameters

The previous sections presented the MEM algorithm which can be used for estimating the model parameters of the process in Figure 4.1. These model parameters are again the priors of the labels in π and the means of the gaussian distributions that the homography parameters are drawn from, μ_k, where each gaussian corresponds to one planar region in the scene. Two implementations were presented for the modified M-Step of the MEM algorithm: one that uses an optimization and another based on a Kalman Filter. Using either implementation, the MEM algorithm presented up to this point is used to estimate all of these parameters, however, in this section methods are presented that can be used to reduce the number of parameters that need to be estimated by the MEM algorithm. These reductions will occur in the parameter vector μ_k.

To begin, we recall the values that μ_k consists of the following parameters listed below:

$$\mu_k = [r, p, y, \varphi_t, \phi_t, \varphi_n, \phi_n, d]^T$$

where r,p, and y are the rotation angles that compose the rotation matrix of the transformation between the cameras, φ_t and ϕ_t are the angles that describe the translation vector of the transformation between the cameras, $\varphi_n$andϕ_n describe the normal vector of the plane and d is the distance from the camera center of the first camera to the plane along the normal described by $\varphi_n$andϕ_n. We note that either

implementation of the MEM algorithm described thus far will solve for all of these parameters for each homography which results in solving for $8K$ parameters where K is the number of planes in the scene.

While solving for $8K$ parameters is correct, it is noted that in many cases this will be unnecessary. The cases that we are referring to are those in which the planes stay static in the scene with respect to one another. This situation arises in many ways such as when the planes belong to a single rigid object such as a building, or when images of the scene are captured by two different cameras simultaneously. When this property is upheld, the parameters that make up the rotation matrix (r, p, y) and the two angles that make up the translation vector (φ_t, ϕ_t) will be the same for every plane in the scene. This results in the number of parameters that must be estimated by the MEM algorithm to decrease to $5 + 3K$ parameters. In terms of the implementations, this reduction of parameters changes how the modified M-Step operates. For the optimization based M-Step this change can be implemented by optimizing the $5 + 3K$ parameters all at once as opposed to K individual optimizations of 8 parameters. The Kalman Filter based method is harder to adapt and would require interleaving the filtering for each planar region's parameters. That is, a point correspondence that has a high posterior probability for u_1 is used to filter the estimate of the parameters φ_n, ϕ_n, d in u_1 and then the point is used to filter the estimate of the global parameters for all of the $u_k's$. The next point would have a high probability for μ_2 and would do a similar update. This would repeat for all $u_k's$.

While this concept offers a reduction in the number of parameters to estimate, it is noted that the optimal solution to the reduction of the number of parameters would be to solve for the motion parameters $(r, p, y, \varphi_t, \phi_t)$ that are common for all the planar regions outside of the MEM algorithm. Recall that the motivation for using MEM to estimate the $u_k's$ for each planar region stems from the fact that this is a problem of incomplete data where the labels of the point correspondences are unknown. However, when the above transformation constraints are upheld, all of the observed point correspondences share the same motion parameters and the problem of not knowing the correspondences' labels becomes irrelevant. This means that

there is no justification for using MEM since the data set can be viewed as being complete. Solving for the motion parameters before the MEM algorithm reduces the number of parameters to estimate from $8K$ parameters down to $3K$. While this reduction of parameters may seem small when compared to the implementation that estimates $5 + 3K$ parameters, we note that one advantage of solving for the motion parameters before the MEM is that we are able to use independent optimizations and/or Kalman Filtering M-Steps again.

In light of this realization, it is noted that the most common method used to estimate the motion parameters between two images is through decomposing the fundamental matrix. Solving for the fundamental matrix is straight forward and can be done using a least squares technique. However, due to noise in the matching, it is preferred that a robust estimation technique, such as RANSAC, is used to estimate the fundamental matrix. This insures that gross outliers will not have an ill effect on the computation of the fundamental matrix. Even though the use of a robust estimation technique is beneficial when dealing with outliers, the robust estimation techniques themselves have shortcomings. Specifically, since RANSAC is based on random sampling, we are not guaranteed that we would arrive at the exact solution for the fundamental matrix of a set of point correspondences if we were to apply RANSAC multiple times. In addition to this, the fundamental matrix that is estimated by the robust estimation technique will always be noisy since techniques such as RANSAC have no means to take into account the presence of noise that exists in the correspondences. The decomposition of the noisy fundamental matrix will produce noisy motion parameters which will compromise the estimation performed by the MEM algorithm since these parameters will be held constant.

Since the presence of noise in the point correspondences will always influence the outcome of the robust estimation techniques, it would be ideal to estimate the motion parameters using a method that takes into account the presence of noise. We recall that this is the exact purpose of the Kalman Filter; that is, estimating the state of a process from noisy measurements. In this case, the state vector the Kalman Filter is estimating consists of the motion parameters, while the noisy measurements are the point correspondences. By employing Kalman Filtering to estimate the motion

parameters, we are able to refine the noisy estimate that is received from a robust estimation technique using the point correspondences while taking into account the noise that exists in the point correspondences. This will result in a more accurate estimate of the motion parameters which can then be used by the MEM algorithm.

In the remainder of this section, the derivation of the necessary Kalman Filter equations is presented. It will be shown that this derivation requires a unique approach to 3D reconstruction which is presented in the next section.

4.7.1 3D Reconstruction Using Rays

Following a similar path of reasoning in Section 4.6.2, the first step in deriving the Kalman Filter equations is to define a constraint equation. One criterion of this constraint equation is that it must utilize the motion parameters to be estimated and the noisy point correspondences. A second criterion is that the constraint equation should produce a value of zero when provided a noiseless measurement along with correct motion parameters. A constraint equation that satisfies this criteria can be defined based on 3D reconstruction. That is, given a point correspondence and the current estimate of the motion parameters, a 3D point can be calculated and then reprojected back onto both images. In the presence of noise and incorrect motion parameters, the reprojected points of the 3D point will not fall on the same location of the image planes as the points used to create the 3D point. The difference between the original points and the reprojected points can be used to define the constraint equation below:

$$f(\mathbf{x'}_n, \eta, \mathbf{x}_n) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$$

where

$$f_1 = (u_1 - u_1^{'})^2 + (u_2 - u_2^{'})^2$$

$$f_2 = (v_1 - v_1^{'})^2 + (v_2 - v_2^{'})^2 \tag{4.27}$$

In these equations, η is a vector that contains the motion parameters $(r, p, y, \varphi_t, \phi_t)$, $u_1$andv_1 are the coordinates of $\mathbf{x}_n$ while $u_2$andv_2 are the coordinates of $\mathbf{x'}_n$. The values $u_1^{'}, v_1^{'}, u_2^{'}$,and $v_2^{'}$ are the coordinates of the reprojected points that are calculated using:

$$\begin{bmatrix} s_1 u_1^{'} \\ s_1 v_1^{'} \\ s_1 \end{bmatrix} = A_1 X \tag{4.28}$$

$$\begin{bmatrix} s_2 u_2^{'} \\ s_2 v_2^{'} \\ s_2 \end{bmatrix} = A_2 [R\, t] X \tag{4.29}$$

where $A_1$and A_2 are the intrinsic parameters of the cameras that took the images, R and t are the rotation matrix and translation vector composed of the motion parameters in η which describe the transformation between the cameras coordinate systems and$X$ is the calculated 3D point using the standard method of 3D reconstruction that is based upon a least squares solution.

Since the constraint in (4.27) is non-linear it is necessary that the equation be linearized. This requires that the partial derivative of the constraint equation be taken with respect to the coordinates in $\mathbf{x'}_n$ and the motion parameters in η. It is here that the short coming of this constraint equation is exposed. First, taking the partial derivative of (4.27) requires that (4.28) and (4.29) get factored into (4.27) and the partial derivative is taken with respect to the resulting expression. It is noted that the difficulty of calculating the partial derivatives of (4.27) is increased due to the squared differences that appear. However, this is necessary since the differences have the possibility of cancelling one another out. As if this task was not daunting enough, we also point out that the reconstruction of the 3D point X also depends on the parameters in η which means that the partial derivatives of the 3D reconstruction step be taken with respect to the coordinates in $\mathbf{x'}_n$ and the motion parameters in η. This partial derivative is impractical because the 3D reconstruction is based on a least squares method not to mention that if such a partial derivative was possible, it would need to be factored into (4.28) and (4.29)

as a substitution for X. These problems inherent to this constraint are too much to overcome. However, the notion of having a constraint equation based on 3D reconstruction is still attractive. This motivates us to define a new method for 3D reconstruction that is easy to differentiate with respect to the motion parameters and coordinates in $\mathbf{x'}_n$.

The method of 3D reconstruction we propose is based on finding the two closest points on a pair of skew lines. Specifically, this method back projects the point correspondence into 3D space by multiplying the individual points by the inverse of the camera calibration matrices. This results in a ray departing from the center each of the two cameras. In the case of no noise, these two rays would intersect in 3D space at the location of the 3D point we are trying to solve for. Unfortunately, in the presence of noise or incorrect motion parameters, these rays will not intersect which makes the problem of determining the 3D point difficult. The standard method of 3D reconstruction essential solves for a point in space that is the minimum distance from both of these rays. Since the point does not lay on either of the rays, it projects onto a different point of the image planes than the rays. This is the reason that there are two errors that are taken into account in the constraint equation in (4.27).

Instead of finding a 3D point that is close to the two rays, our method of 3D reconstruction focuses on finding the two closest points of the rays and using one of them as the reconstructed 3D point. In order to find the two closest points on the rays, we must first describe the rays using a vector form such as:

$$l = p + sd \tag{4.30}$$

where l is the line, p is the origin of the line, d is the line's direction vector, and s is a magnitude value. When the points in the correspondence are back projected into space, we receive two direction vectors: d_1 and d_2. At this point, the lines that these direction vectors describe are:

$$l_1 = s_1 d_1$$

$$l_2 = s_2 d_2$$

We point out that the above line equations differ from (4.30) in that they pass through the origin of their respective camera's coordinate systems so $p = [0, 0, 0]^T$. However, in order to be able to calculate the minimum distance between these rays, we must define them in the same coordinate system. By assigning the coordinate system of the first camera to be our reference coordinate system, the equations of the rays become:

$$l_1 = s_1 d_1$$

$$l_2 = t + s_2 R d_2$$

We note that the equation for the first ray remains the same since it passes through the origin of our reference coordinate system. The second ray's equation, however, has been modified by the rotation matrix R and translation vector t that define the transformation between the two cameras. Using these equations, we can then solve for the values of $s_1$and s_2 that describe the two closest points on the rays. In this work, we then define the reconstructed 3D point as being:

$$X = s_1 d_1$$

One important property to note about this form of 3D reconstruction is that the 3D point will project back onto the exact location of the original point for one of the images. This means that all of the error that (4.27) was attempting to capture from two images is now contained in one of the images. This allows us to define a new constraint equation:

$$f(\mathbf{x'}_n, \eta, \mathbf{x}_n) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$$

where

$$f_1 = u_2 - u_2^{'}$$

$$f_2 = v_2 - v_2^{'}$$

and

$$\begin{bmatrix} s_2 u_2^{'} \\ s_2 v_2^{'} \\ s_2 \end{bmatrix} = A_2[R\,t]X$$

It can be seen that using this method of 3D reconstruction significantly simplifies the constraint equation. In addition to this, it is noted that it is possible to differentiate the 3D reconstruction method with respect to the motion parameters and the second point in the point correspondence. This equation is the one that will be used to derive the Kalman filter equations in the next section.

4.7.2 Kalman Filter Derivation

The previous section presented a constraint equation that is linearizable. Using that equation, the necessary Kalman Filter equations can be derived. Linearization of this equation results in the following:

$$f(\mathbf{x'}_n, \eta, \mathbf{x}_n) = f(\hat{x}'_n, \hat{\eta}, \mathbf{x}_n) + \frac{df}{dx_n}(\mathbf{x'}_n - \hat{x}'_n) + \frac{df}{d\eta}(\eta - \hat{\eta}) = 0 \tag{4.31}$$

As with the homography constraint equation, this equation can be rearranged into the Kalman Filter measurement equation where:

$$y_k = -f(\hat{x}'_n, \hat{\eta}, \mathbf{x}_n) + \frac{df}{d\eta}\hat{\eta}$$

$$M = \frac{df}{d\eta}$$

$$u_k = \frac{df}{dx_n}(\mathbf{x'}_n - \hat{x}'_n)$$

The error covariance U is once again defined as:

$$U = \frac{df}{dx_n}\Sigma_x\frac{df}{dx_n}^T$$

where $\Sigma_x$is the error covariance from the matching algorithm in Figure 4.1. These equations can then be used to derive a similar set of update equation as in (4.26).

Using the standard Kalman Filter update equations, the estimate of the motion parameters returned by a robust estimator can be refined using all of the collected point correspondences. This allows us to take into account the error in the point correspondences when estimating the motion parameters. This results in a more accurate set of motion parameters for the MEM algorithm to use while reducing the number of parameters that the MEM algorithm must estimate.

4.8 Case 3: Single Plane Detection

The last topic we discuss in this chapter is the application of the MEM algorithm to a single plane environment. All the previous discussions and proposed implementations of the MEM algorithm up to this point have assumed that multiple planes exist in the environment and the point correspondences collected between the images belong to one of the planar regions in the scene. While multiple plane estimation is an important task, extending the MEM algorithm to estimate the parameters of a single plane is equally important since there exists many algorithms that can benefit from such estimation, specifically algorithms that rely on ground plane detection.

The difficulty of extending the MEM algorithm to estimate the parameters of a single homography comes from the fact that in this type of problem outliers will be encountered. That is, if all of the point correspondences belonged to the single planar region whose homography parameters are being estimated, there would be no reason to employ the MEM algorithm since the labels of the points would be known (i.e. they belong to the only plane in the scene). The outliers encountered in this

estimation problem are any point correspondences whose 3D point in space does not lay on the single plane. The presence of these outliers requires that the standard MEM framework be modified in order to take them into account. Unfortunately this modification is not as straightforward as one would hope.

We recall that in the E-Step of the MEM algorithm the posterior probabilities of each label given the point correspondence is calculated for all of the correspondences. These posterior probabilities are calculated using the priors for each label and the likelihood of the statistical parameters associated with the label for a given point correspondence. In a single plane setting only two posterior probabilities are calculated in the E-Step: the posterior probability for the correspondence being labeled as belonging to the plane and the posterior probability of the correspondence being labeled as an outlier. This requires that two likelihoods must be calculated.

The first likelihood that must be calculated is the likelihood of the statistical parameters of the homography. Calculating this likelihood is done using the same equations as before in the multiple plane MEM algorithm. On the other hand, it is noted that calculating the likelihood of an outlier cannot be done in the same way because there exist no statistical parameters associated with the outlier label. Because of this, we must resort to assigning an outlier likelihood value that is based on the likelihood of the homography. Specifically, the likelihood value assigned to a correspondence that represents the outlier likelihood value is $1 - p(\mathbf{x}'_n|\mu_k, \mathbf{x}_n, \Sigma_x)$ where the latter value is the homography likelihood. That is, if the likelihood of the homography is 1, then the outlier likelihood is equal to 0 and vice versa. By assigning likelihood values in this manner, we are able to calculate the necessary posterior probabilities in the E-Step.

This assignment of likelihood values is the only difference between an implementation of MEM that estimates the parameters of a single plane and an implementation that estimates the parameters of multiple planes. The modification steps remain the same in both cases where the statistical parameters of the homography are updated using the point correspondences. The effectiveness of this variation of MEM will be tested in the following chapter where the algorithm is applied to ground plane detection and is compared with other single model estimation techniques.

4.9 Variations of MEM

In this chapter we presented the MEM algorithm which can be used to estimate the parameters of multiple planar regions in a scene or to estimate the parameters of a single planar region in the presence of outliers. In this presentation, many different variations of the algorithm were mentioned that either vary the number of parameters the algorithm estimates or the type of methods used in the modified M-Step. While all implementations are valid, it is noted that there exist variations of the algorithm that offer significant advantages over others in terms of convergence and speed. For instance, a variation of MEM that only estimates the normal angles and distance value will obviously converge quicker than a variation that estimates all eight parameters. Furthermore, it is noted that variations of the MEM algorithm exist that have not been mentioned in this chapter, such as an MEM algorithm variation whose M-Step incorporates genetic algorithms. Despite these implementation differences, all of these implementations still conform to the EM framework where a set of parameters are being estimated using incomplete data.

Chapter 5

Results

5.1 Modeling the Error Covariance: Σ_x

In Section 4.2 a model was presented that describes the underlying process which produces the point correspondences that are observed in a planar detection problem. This model takes into account the noise observed when collecting point correspondences using a feature matching algorithm. In the model, this noise is assumed to be normally distributed with a mean of 0 pixels and a covariance of Σ_x. During the derivation of the MEM algorithm it was assumed that these parameters are constant and that the value of Σ_x could be determined empirically, thus not needing to be estimated by the MEM algorithm. When deriving the Kalman Filter based Maximization Step in Section 4.6.2 and the Kalman filter based algorithm for estimating motion parameters in Section 4.7, Σ_x reappears and is once again assumed to follow a Normal distribution. The purpose of this experiment is to investigate the error obtained when using a feature matching algorithm to find point correspondences in an image pair, specifically, to see if this error follows a normal distribution as it is assumed in this body of work.

In this work, feature matching between a pair of images is done by first detecting interest points in both images. Interest point detection is done by applying the difference of Gaussians function to different scales of the images. The resulting set of interest points are then described using the SIFT descriptor. Some of the attractive features of the SIFT descriptor is its invariance to scale and its immunity to small

affine transformations. While the above algorithm is used in many applications for feature matching, the set of match points returned is not perfect. That is, very rarely will a set of points be an exact match. The question then becomes: Can the error be modeled using a normal distribution, and if so what are the approximate statistics that define the distribution? By approximate statistics, we are referring to finding a rough neighborhood of values for the standard deviation of the error such as 50 pixels, 20 pixels, or less than 10 pixels.

In order to find an empirical solution to that question, 2000 correspondences were collected from images of both outdoor and indoor scenes. The ground truth was then collected for these correspondences by hand in the following manner. First, assistance was provided by students who had no background in computer vision in an attempt to provide an unbiased collection of ground truth. Each student was assigned a subset of the 2000 correspondences. To collect the ground truth for a correspondence, the student would be presented with the two images that the correspondence was obtained from. One of the images would have the correspondence marked and the student was instructed to select the location of the point in the other image. This method was used for all 2000 correspondences and yielded the following results:

$$\mu = \begin{bmatrix} 0.215 \\ 0.400 \end{bmatrix} \quad \sigma_\mu = 4.9057 \quad \sigma_v = 4.9079$$

The mean obtained from this method was on the subpixel scale while the standard deviations in the u and v directions were both approximately 5 pixels. These results support the assumption that the correspondence error follows a Normal distribution with a mean of 0 pixels. It is important to note again that the values of σ_μ and σ_v collected from this experiment provide a rough estimate of the distribution's parameters and not necessarily the exact values that need to be used by the MEM algorithm. That is, these results indicate that using standard deviations of 3 pixels for the MEM algorithm is reasonable, while standard deviations of 15 or 20 pixels might not be accurate estimates. In fact there will be instances of using the MEM algorithm where the values assigned for these standard deviations affect the classi-

fication. The influence of these parameters on the classification results obtained by the MEM will be discussed further in Section 5.3.2.

This experiment confirms our assumption that the noise observed when collecting point correspondences follows a Normal distribution with a mean of 0 pixels. This validates the model presented in Section 4.2 which the MEM algorithm was derived from. These results also justify the use of the Kalman filter in our Kalman filter based Maximization Step and the algorithm that refines the estimate of the motion parameters.

5.2 Ground Plane Detection For Mobile Robot

The first application of the MEM algorithm presented is a mobile robot navigation application where MEM is used to implement a simple target tracking and following algorithm. In this application, MEM is used to determine which point correspondences belong to the target the mobile robot is following by classifying points as belonging to the ground plane or the target. In order to focus on the performance of the classification using MEM, the tracking algorithm was simplified by assuming that the mobile robot only sees one object in its view, that is, all of the points that are not classified as belonging to the ground plane are assumed to belong to a single target. The tracking and following algorithm consists of four steps: 1) Initialization; 2) Feature Detection and Matching; 3) Ground Point Classification; and 4) Robot Motion Control which are illustrated in Figure 5.1.

The purpose of the initialization step of the target follower algorithm is to determine a set of homography parameters, θ , that can be used as an initial estimate for MEM. Obtaining this set of parameters is important for two reasons. First, due to the nature of the EM algorithm an initial estimate of the model parameters that are being estimated by the algorithm need to be provided. For our MEM algorithm, these model parameters are the homography parameters, θ, and prior probabilities of the labels, π. An initial estimate for π is trivial to provide since we will assume that the prior probabilities for the labels are equivalent. An initial estimate for θ, however, is more difficult to provide since we do not know roughly where the

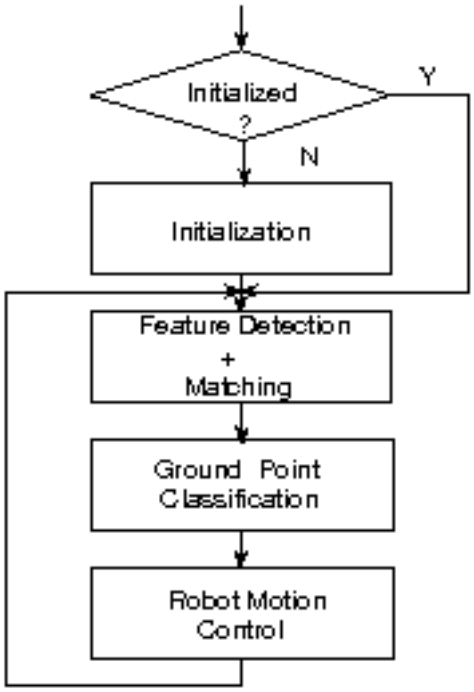


Figure 5.1: Flow chart for tracking and following algorithm

plane is in the scene. Second, since the homography constraint can be applied to any plane in the environment, it is necessary that the algorithm be initialized with respect to the desired plane, in this case the ground plane. Since this initial estimate does not need to be the exact set of parameters that describe the ground plane, the initialization step can be kept simple. The initialization step used in this experiment required that the algorithm be provided with two overlapping images of the ground plane (with no target in the scene) taken by the mobile robot's on board cameras. The initialization step then finds point correspondences between the images and runs the MEM algorithm with the initial assumption that all of the point correspondences belong to the ground plane. It is important to stress the fact that this initialization only needs to be done once for any robot, that is, for any configuration of the cameras.

Once the robot has been initialized, the algorithm proceeds to the Feature Detection and Matching step. This step begins with the capture of two images. Point correspondences are then detected between the images using SIFT [12]. This set of point correspondences should belong to both the ground plane of the scene and

Algorithm 5.1 MEM Algorithm

```
Input: Point Correspondences, Homography Parameter Estimate
Output: Classified Correspondences
  while(!termination_criteria_met)
    Calculate_Probabilities()
    Update_Homography_Parameters()
    Update_Class_Probabilities()
  end
  Calculate_Probabilities()
  Assign_Clustering()
```

the target. It is this output of point correspondences that is passed onto the MEM algorithm for clustering.

After the correspondences have been detected, the next step is Ground Point Classification. In this step, the MEM algorithm is used to classify the correspondences as belonging to ground plane or the target as described in 5.1.

From 5.1 it can be seen that the input to the MEM algorithm is the set of point correspondences obtained in the previous step and an initial estimate for the homography parameters. For the implementation presented here, the estimate of homography parameters provided to the algorithm is the one that was obtained in the initialization step. Another alternative for providing an initial estimate to the MEM algorithm would be to use the set of converged parameters from the previous image pair. It is here that we note that using an estimate based on the initialization step or the previous set of converged parameters is valid because our implementation uses a stereo camera setup. This implies that the values of θ should remain fairly consistent for each pair of images since the cameras remain fixed with respect to each other. However, this does not mean that MEM cannot be applied to a single camera setting. In a single camera setting, the initialization step is still necessary in that it provides an estimate for the parameters that describe the normal vector of the plane which remains consistent for each pair of images. Estimates for the six remaining parameters can then be determined on a per image pair basis using dead reckoning or the decomposition of the fundamental matrix. By using two fixed cameras, the step of creating an initial guess is simplified in that we can directly use the estimate from the initialization step for every pair of images. Regardless of

which setup is used (single or multiple cameras), classification using MEM remains the same.

Once the MEM algorithm has returned the classification, the correspondences that are not classified as belonging to the ground plane are used to track the target in the Robot Motion Control step. The goal of this step is to keep the tracked target in the center of one of the cameras' views. This is done by calculating a heading value to the target by back-projecting the target correspondences into 3D space. This results in a number of rays departing from the camera which are then projected onto the horizontal axis of the camera. The accumulation of intersection points between the rays and the horizontal axis are used to calculate a heading to the target. The mobile robot corrects itself by this heading so that the target is once again in the center of the camera's view.

The proposed target tracking and following algorithm was implemented and tested on a HP Pavilion dv6 running Intel(R) Core(TM)2 Duo CPU @ 2.0GHz. In order to improve the performance speed of the navigation algorithm, the MEM step ran once for every four image pairs collected. That is classification of the point correspondences was calculated every time the image pairs were collected, but the MEM refined the homography parameters every four image pairs. For the experiment, two P3DX mobile robots from Mobile Robots Inc. were used with one of the mobile robots running the proposed tracking and following algorithm. The second robot served as the target robot and executed a program that allows the robot to "wander" through the hallway while avoiding collisions using its on board SONAR ring. The experiment consisted of capturing 6 trials of the "wandering" robot running all the way down the hallway while the second robot followed using the tracking and following algorithm. This equated to the collection of 1000 frames with each frame containing roughly 400 correspondences.

The statistics collected for the experiment are summarized in Table 5.1. Since no ground truth was available for the data the ground truth had to be manually obtained. For this reason, 20% of the 1,000 frames were randomly sampled from the six sequences. The table summarizes the statistics of these 20% of the samples.

As it can be seen in the table, the subset of randomly selected frames equated

	Total number of pixels Classified	Correct Classification Percentage	Incorrect classification from SIFT	Incorrect classification from MEM alone
Ground	88,145	99.62%	270 (0.3%)	71 (0.08%)
Non-ground	7,930	99.4%	24 (0.3%)	22 (0.3%)
Total	96,075	99.6%	294 (0.3%)	93 (0.1%)

Table 5.1: Statistics of the Classification

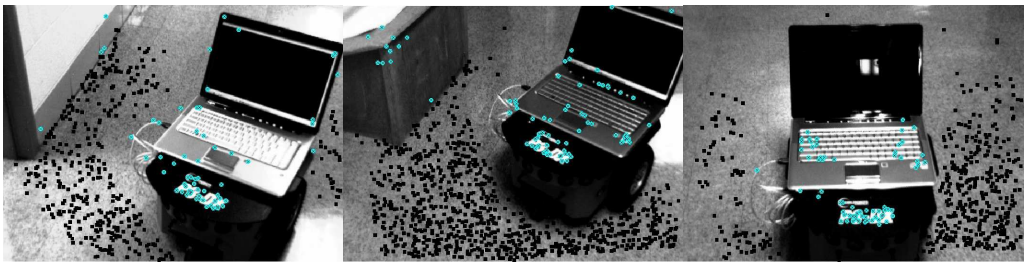


Figure 5.2: Sample images from test sequences. Black squares are pixels classified as ground. Blue circles are pixels classified as non-ground.

to 96,075 correspondence classifications, with 88,145 of these being classified as belonging to the ground plane, and 7,930 belonging to the target. Also, the algorithm returned 341 ground pixels that happened to be misclassified as object pixels. At the same time, it also returned 46 object pixels that were misclassified as ground pixels. However, not all of these misclassifications were due to the proposed algorithm alone. That is, 270 of ground pixels were misclassified because the feature matching algorithm was not able to find a correct match between the pair of images. Similarly, 24 of the target object pixels were misclassified for the same reason. In the end, only 71 of the ground pixels and 22 of the target object pixels were misclassified by the MEM alone. Overall, the MEM algorithm achieves a total correct classification rate of 99.6%. Some samples from the sequences can be seen in Figure 5.2.

5.2.1 Comparison With Other Planar Detection Algorithms

For comparison, the same randomly sampled images were classified using both the RANSAC algorithm and JLinkage algorithm. As with MEM, these algorithms have parameters that can be varied to obtain different results. In order to provide an unbiased comparison between algorithms, Table 5.2 presents the precision and re-

call values for the three algorithms for three different parameter settings. For the RANSAC algorithm, the parameter that is swept is the threshold t_{cs} that determines whether a point correspondence belongs to the consensus set. Low threshold values run the risk of excluding too many correspondences from the consensus set, while high threshold values risk including too many correspondences. Both scenarios can result in a poorly fit model. Since JLinkage has a RANSAC-like sampling step, it also contains the same threshold parameter that must be swept. In addition to this, JLinkage has a second parameter which is the number of hypotheses that must be computed. The table contains results for two different JLinkage algorithms, one that uses 500 hypotheses and another that uses 5000 hypotheses. For the MEM algorithm, the parameter that can vary is the value of Σ_x. Variations in this parameter directly affect the probabilities that are assigned to correspondences and is discussed in more detail in Section 5.3.2. This parameter was swept with values that coincide with the results obtained in Section 5.1. In the table, results are listed for two different versions of the MEM algorithm. The first version labeled with the suffix "_optim" is the version of MEM which uses the optimization for the Maximization step. This is the version of MEM whose results are listed in Table 5.1. The other version of MEM listed in the table is the one that uses Kalman Filtering for the Maximization step and has the suffix "_Kalman".

When it comes to the overall performance of the algorithms, one can see that both versions of the MEM algorithm achieve classification results that are competitive with the JLinkage and RANSAC algorithms. One interesting relationship in the table is the one between the precision and recall values for each algorithm and the parameter being varied. For the JLinkage and RANSAC algorithms, it is observed that as the consensus set threshold increases, the precision of the algorithm decreases while the recall value increases. This is expected because this threshold is a direct measure of certainty for allowing a point correspondence into the consensus set. By setting the threshold low, only correspondences that we are very certain to belong to the ground plane were allowed in the consensus set. This would result in a low recall value because not all of the ground correspondences are being included in the consensus set, while the precision of these correspondences are high since we are only

	Precision	Recall
MEM_optim $\sigma_x = \sigma_y = 2$	0.9989	0.9841
MEM_optim $\sigma_x = \sigma_y = 3$	0.9983	0.9972
MEM_optim $\sigma_x = \sigma_y = 4$	0.9980	0.9972
MEM_Kalman $\sigma_x = \sigma_y = 2$	0.9992	0.9091
MEM_Kalman $\sigma_x = \sigma_y = 3$	0.9986	0.9676
MEM_Kalman $\sigma_x = \sigma_y = 4$	0.9981	0.9842
JLinkage_500 $t_{cs} = 1$	0.9997	0.4864
JLinkage_500 $t_{cs} = 3$	0.9986	0.9952
JLinkage_500 $t_{cs} = 5$	0.9970	0.9955
JLinkage_5000 $t_{cs} = 1$	0.9994	0.5336
JLinkage_5000 $t_{cs} = 3$	0.9982	0.9964
JLinkage_5000 $t_{cs} = 5$	0.9978	0.9969
RANSAC $t_{cs} = 0.0005$	0.9991	0.9268
RANSAC $t_{cs} = 0.005$	0.9979	0.9846
RANSAC $t_{cs} = 0.1$	0.9837	0.9920

Table 5.2: Precision and Recall values for various parameter settings

allowing correspondences that we are very certain of into the set. As this threshold increases, more correspondences that belong to the ground plane are allowed into the consensus set which results in an increase in the recall value. However, this threshold increase also will allow correspondences that do not belong to the ground plane into the consensus set which results in a decrease in precision.

The MEM algorithms also display a similar behavior for the parameters used to construct Σ_x. This relationship exists because Σ_x is used to calculate the likelihoods for the point correspondences in the E-Step, where an increase in Σ_x increases the likelihoods assigned to the correspondences. This directly affects the update of the model parameters in the M-Step since these likelihood values are used to calculate the posterior probabilities which weight the correspondences. That is, when Σ_x is small, only correspondences with small geometric error will be assigned high probabilities and have a significant effect on the update of the homography parameters. Increasing the values used to construct Σ_x will assign higher probabilities to all of the correspondences which allows more correspondences to have an effect on the update of the model parameters. However, setting Σ_x too high will risk using non ground plane correspondences to update the model parameters so caution must be taken when assigning Σ_x.

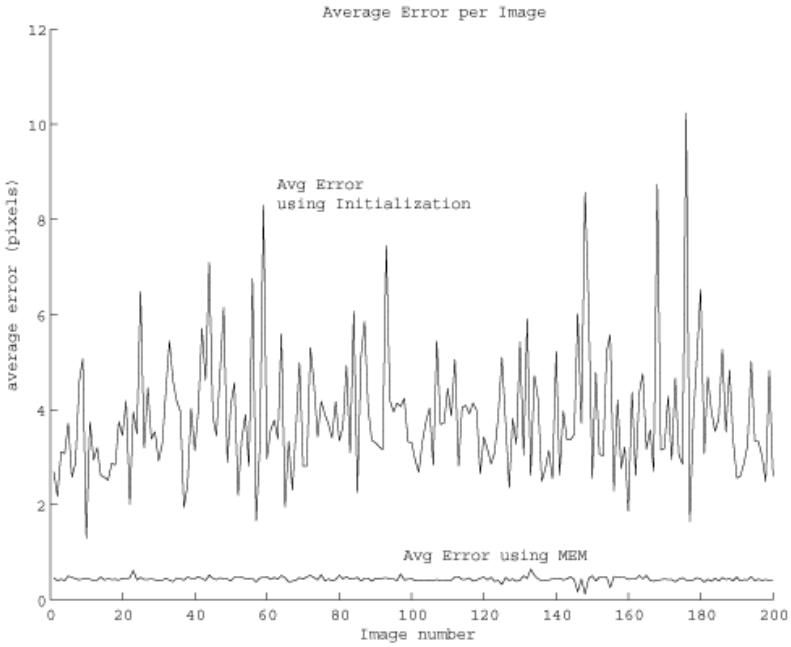


Figure 5.3: Average ground plane correspondence error per image

5.2.2 Initial Estimate Refinement

The purpose of this section is to address the speculation that some readers might have with the effectiveness of the MEM algorithm in this application. We recall that in this experiment, a stereo camera setup was employed which allowed us to utilize an initial estimate that was learned in our initialization step. The argument might be made that if the homography parameters that describe the ground plane are learned in the initialization step and a stereo camera setup is used, could these parameters be used directly to classify the correspondences. This would be analogous to a stereo camera setup that is used for 3D reconstruction where calibration is done once for the setup and the calibration results are used for all images captured. To address this argument we present the plot in Figure 5.3.

The plot in Figure 5.3 was generated by taking the ground plane correspondences for each image pair and calculating their average geometric error using the estimate from the initialization step and the set of converged homography parameters obtained by MEM for that image. This allows us to visualize how well each set of parameters describes the ground plane for the image pair. In the graph it can be seen that the average error per image pair is on the sub pixel level when

the geometric error is calculated using the converged homography parameters from the MEM algorithm. This indicates that the converged parameters returned by the MEM algorithm accurately describe the homography that is induced by the ground plane for that image pair. On the other hand, it can be seen that the average error per image pair varies wildly when calculating the geometric error using the estimate from the initialization step, which means that this estimate does not exactly reflect the homography of the ground plane. Something to note is that the plot using the initial estimate never obtains an average error that is on the subpixel magnitude. This means that for every image pair, MEM is refining the initial estimate to find the true set of parameters that describe the ground plane's homography.

From the plot it is easy to see that the MEM algorithm does have an effect when estimating the homography parameters of the ground plane. However, we would like to illustrate one more advantage that MEM has over using a method that uses the initial estimate for classification. If one were to strictly use the estimate obtained from the initialization step for classification, one problem they would encounter would be to determine a threshold to set for classification. From the graph it can be seen that this threshold could vary from less than 2 pixels to 10 pixels and be a valid threshold for many of the image pairs. However, there is always a consequence when setting a threshold for such a method. Setting the threshold too low will result in ground correspondences not being correctly classified, while a threshold that is too large runs the risk of misclassifying non ground plane correspondences as belonging to the ground plane. By using MEM, we eliminate the need to set a classification threshold. We recall that the purpose of MEM is to estimate the model parameters θ and π which is illustrated in the graph. Once these parameters are learned, the posterior probabilities for all of the correspondences can be calculated and the correspondences can be classified based on which class they have the highest posterior probability.

5.3 Single Plane Tracking

The previous section presented an application that employed MEM in a target tracking and following algorithm for a mobile robot. In that application, the initial estimate of homography parameters used for the MEM was learned ahead of time using the proposed initialization step. As mentioned before, using this learned initial estimate for every image pair is valid because the mobile robot platform uses two cameras for its navigation. Using two cameras ensures that the transformation undergone between image pairs should be fairly consistent. In this section we present an application where MEM is used to classify point correspondences between image pairs of an image sequence captured by a single uncalibrated camera.

The application presented in this section illustrates the use of MEM in a plane tracking scenario. In this application MEM is used to track the ground plane of a scene through a sequence of airborne video. The tracking of the ground plane can be a useful preprocessing step for a target tracking and geolocation algorithm such as the one in [1]. Specifically, tracking the ground plane can be used to filter out correspondences that belong to objects that are of no interest, such as buildings. Using MEM in a tracking scenario is logical because in many tracking algorithms it is important to use prior information about the target in previous images to predict its location in the next image. By using MEM for tracking the ground plane throughout an image sequence, we are able to make use of the prior knowledge we have of the ground plane from previous images in the sequence. This is an advantage that MEM offers over other planar detection algorithms such as RANSAC and JLinkage which may not incorporate prior knowledge into their standard frameworks.

While this application demonstrates the use of MEM in a plane tracking scenario, it also demonstrates the use of MEM in a setting that differs from the mobile robot application. First, this application uses MEM on an outdoor image sequence where the ground plane to be detected is the ground plane of the earth which may not always be a smooth plane. Second, the sequence of images in this application is captured by a single uncalibrated camera. Using a single camera means that the initial estimate used for each classification cannot be learned ahead of time since the transformation undergone between pairs of images cannot be guaranteed to be

consistent. Instead of using the same initial estimate for every image pair we will use the previous knowledge of the plane in earlier image pairs to derive an initial estimate.

In this method, we take the previously classified image pair which consists of images: I^{t-1} and I^t, and find the point correspondences that are in common with the new image pair that is to be classified which consists of images: I^t and I^{t+1}. These common correspondences and their posterior probabilities –which were calculated for the previous image pair– are used in an optimization step, which is similar to our Maximization Step, which generates an initial guess for the parameters in θ_t. Once this initial guess has been created, the algorithm proceeds as described in Algorithm 5.1.

While this method provides a means to provide an initial estimate based on the previous frames, we note that an initialization needs to be performed for the beginning of the sequence. Due to the data not containing any images of purely ground plane pixels, the initialization for the sequences had to be assigned manually. After the first pair of images has been classified, the application automatically generates a estimate for the next pair of images using the previously described method.

This application of the MEM algorithm was tested on a data set which consisted of three sequences of airborne video of an urban area. These three sequences combined provided roughly 100 image pairs containing over 100,000 point correspondences. These sequences also provided a range of different scenarios to test the MEM algorithm: Sequence 1 is composed of images that contain a large amount of ground plane and few buildings, making it the easiest of the three sequences for classification. Sequence 2 contains images taken of a downtown area with many large buildings and less area that is ground plane. Sequence 3 contains many buildings like Sequence 2 but, the buildings are typical of a residential area meaning they are much smaller in size. As with the previous experiment, the ground truth for this data was obtained manually.

This data set was used to test the optimization version of the MEM algorithm, the JLinkage algorithm and the RANSAC algorithm. As before, each algorithm was run multiple times for different values of their parameters. The table below reports

	Sequence 1 Precision	Sequence 1 Recall	Sequence 2 Precision	Sequence 2 Recall	Sequence 3 Precision	Sequence 3 Recall
MEM_optim $\sigma_x = \sigma_y = \sqrt{2}$	0.8396	0.9944	0.8678	0.9608	0.7822	0.9720
MEM_optim $\sigma_x = \sigma_y = 2$	0.8096	0.9981	0.7765	0.9659	0.7337	0.9922
MEM_optim $\sigma_x = \sigma_y = 3$	0.7965	0.9985	0.6969	0.9949	0.7049	0.9949
JLinkage_500 t_{cs}= 2	0.8836	0.4958	0.9095	0.5920	0.8540	0.5431
JLinkage_500 t_{cs}= 4	0.8572	0.8763	0.8243	0.8410	0.7810	0.8889
JLinkage_500 t_{cs}= 6	0.8109	0.9387	0.7805	0.9073	0.7370	0.9588
JLinkage_500 t_{cs}= 9	0.7962	0.9986	0.6903	0.9452	0.7039	0.9972
RANSAC: threshold = 0.001	0.8485	0.9311	0.7786	0.8922	0.7694	0.9191
RANSAC: threshold = 0.01	0.7988	0.9867	0.6788	0.9798	0.6940	0.9813
RANSAC: threshold = 0.1	0.7948	0.9956	0.6524	0.9917	0.6820	0.9918

Table 5.3: Statistics of the Classification Compared with RANSAC and JLinkage

the precision and recalls for each configuration of algorithm parameters.

Once again, the relationship between parameter values and precision and recall outputs can be noticed in Table 5.3, where the increase in parameter value corresponds to an increase in recall and a decrease in precision. From the table it can be seen that both the MEM and RANSAC algorithms appear to be less sensitive to the change in their parameter values and consistently maintain high recall values. JLinkage, however, is much more sensitive to its parameter change and requires a threshold of 9 pixels to be comparable with the worst case results of the MEM and RANSAC algorithms.

Overall, it can be seen that the MEM algorithm with $\sigma_x = \sigma_y = \sqrt{2}$ achieved the best results of all the tests. This configuration of MEM was able to achieve the highest precision values for recall values over 95 percent. The JLinkage algorithm did have configurations where it reported high precision values, but this came at the cost of recall values that were much lower in comparison to the MEM and RANSAC

Figure 5.4: Sample images from Sequence 3.

algorithms. For Sequence 2, MEM outperformed both RANSAC and JLinkage by achieving precision values over 75% while maintain recall values over 95%. For Sequence 1, MEM and RANSAC achieve comparable results, while JLinkage reports results that are comparable with the worst results of MEM and RANSAC. The same can be said for Sequence 3, however, all algorithms performed worse on this data set than Sequence 1. The reason for this decrease in performance comes from the nature of the sequence. As previously mentioned, Sequence 3 contains residential buildings which sit much lower to the ground and are hard to differentiate from the ground plane by the algorithms. Examples of classification by the MEM algorithm for Sequence 3 are provided in Figure 5.4.

The sample images in Figure 5.4 show classification results for the MEM algorithm that used $\sigma_u = \sigma_v = \sqrt{2}$. In these images it can be seen that the buildings that sit a few stories tall are rarely misclassified, while some of the smaller buildings, such as houses, are being classified as belonging to the ground plane. The reason for these misclassifications comes from the fact that the baseline between the two images is not large enough to determine a distinction between the ground

plane of the earth and a house whose roof may only be 25 feet above the ground plane. That is, the image sequence was taken from a high altitude where 25 feet is a relatively small distance. The perception of depth will increase as the baseline between the cameras increases, which means that images that were taken close to the same location will have little depth perception. Unfortunately, as the baseline between images increases, the area of overlap between the images decreases which results in less point correspondences being detected between an image pair.

From this experiment we can see that MEM achieves competitive, if not the best, results when compared to the JLinkage algorithm and RANSAC in a tracking scenario. We note that the success of the MEM algorithm in this scenario comes from the ability of the algorithm to use the prior information about the plane in the previous image pairs. This use of prior information is done through the formation of the initial estimate that MEM requires which is discussed in the next section.

5.3.1 Importance of Initial Estimate

Sequence 2 is a data set that provides a separation of the algorithms based on performance and illustrates the importance of using MEM for plane tracking. For this sequence, the configuration of MEM with$\sigma_x = \sigma_y = \sqrt{2}$ clearly outperforms the best implementations of RANSAC and JLinkage. The reason for this comes from the difficulty of the data set and the advantage that MEM has over other planar detection algorithms by being able to use prior knowledge of the scene. Specifically, the sequence contains many images where the ground plane may not be the dominant plane. One such example is shown in Figure 5.5 where the tops of the buildings are relatively close to the same height and form a planar region.

In this situation, algorithms such as JLinkage and RANSAC will tend to locate the dominant plane in the scene which, for this scene, may be the tops of the buildings. MEM has a better chance of overcoming this problem by incorporating previous knowledge about the scene to form an initial estimate that reflects the ground plane's position in previous scenes. This results in an accurate classification of correspondences like that shown in Figure 5.5. Examples of classifications using RANSAC and JLinkage for the scene are provided in Figure 5.6.

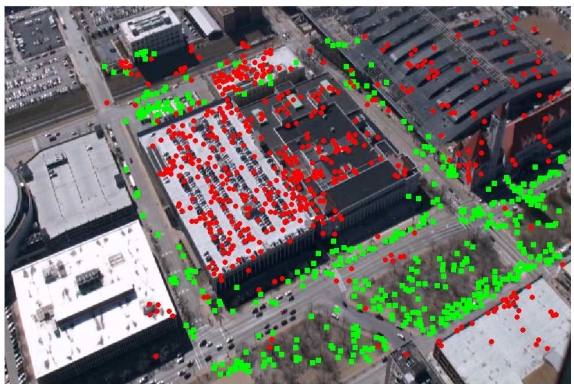
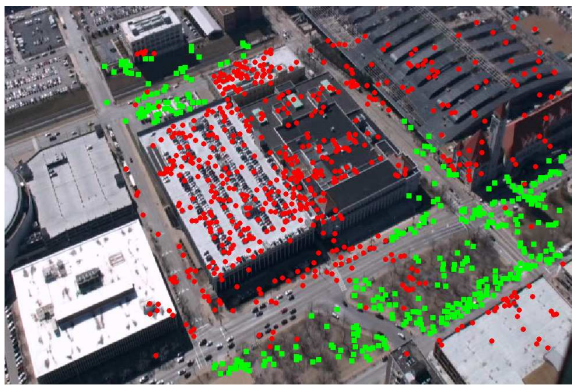
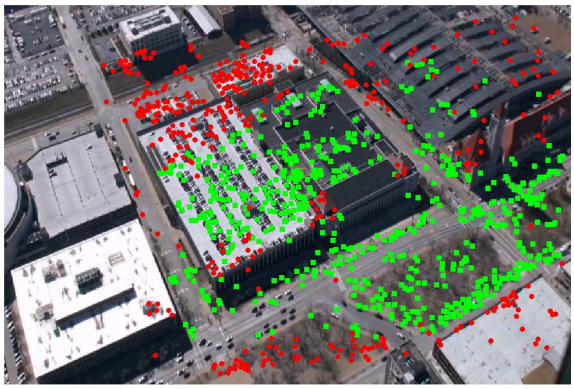


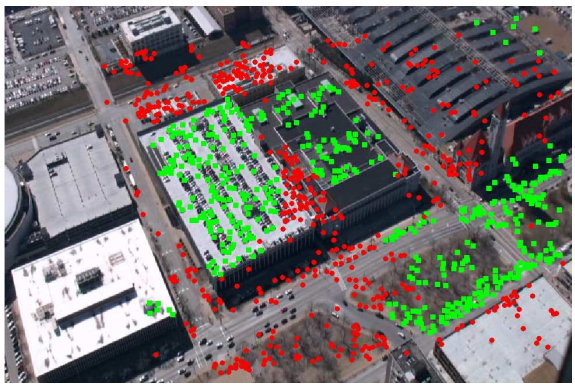
Figure 5.5: Example where dominant plane is not the ground plane



(a) Classification Using JLinkage $t_{cs} = 4$



(b) Classification Using RANSAC $t_{cs} = 0.001$



(c) Classification Using JLinkage $t_{cs} = 2$

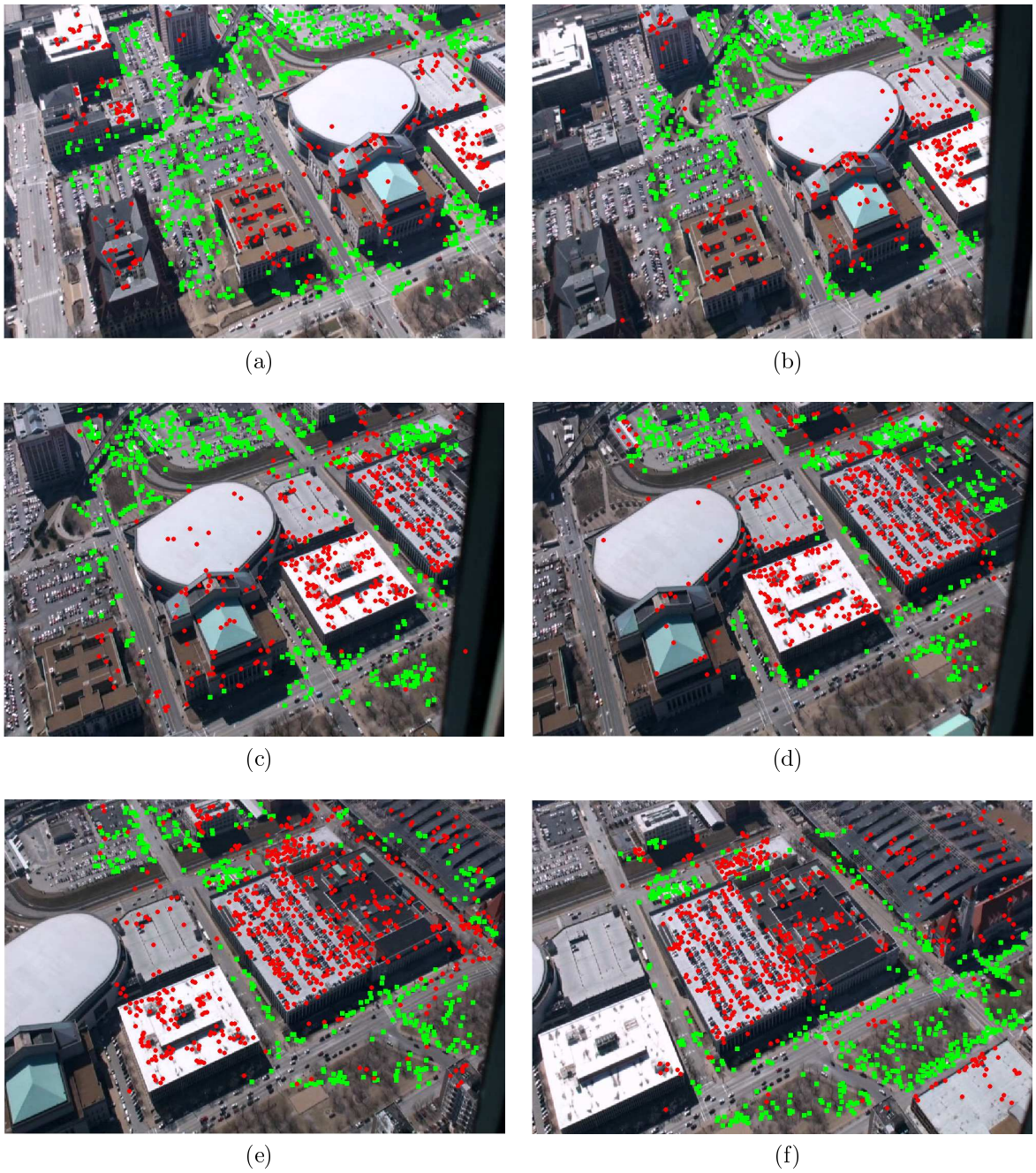
Figure 5.6: Classification results for RANSAC and JLinkage for ground plane detection

Figure 5.6 provides examples of when RANSAC and JLinkage both detect the tops of the buildings as the ground plane. This is because these algorithms are being used to detect the dominant plane in the scene, which for this scene is not clearly the ground plane. Also in the figure an example is shown where JLinkage is able to achieve a satisfactory classification of the ground plane. However it is noted that this classification was achieved for the configuration of JLinkage which uses $t_{cs} = 2$ which achieved poor results overall. This is obviously a configuration of the JLinkage algorithm that would not be ideal to use for the sequence.

While the use of an initial estimate can be powerful in a tracking application, it can also have an adverse effect on the classification. The example we present comes from the same scene in Sequence 2 when the MEM algorithm has a configuration of $\sigma_x = \sigma_y = 4$. To illustrate this point Figures 5.7 and 5.8 provide two subsequences of Sequence 2, one where the configuration of MEM is $\sigma_x = \sigma_y = 2$ and the other where $\sigma_x = \sigma_y = 4$.

We start by looking at the image sequence in Figure 5.7 which corresponds to the configuration of MEM where $\sigma_u = \sigma_v = \sqrt{2}$. In this sequence the tracking of the ground plane can be observed and one can see that the tracking does diverge a small amount in the fourth image where it can be seen that some of the building top is classified as the ground plane. However we are quick to note that this misclassification is overcome in the next image where a satisfactory classification is observed. This is due to the fact that while a part of the building was misclassified in the fourth image, a large amount of the ground plane was correctly classified, so our initial estimate that was created to classify the fifth image was based on many correspondences that belong to the ground plane. This initial estimate was accurate enough that the MEM algorithm was able to converge to the ideal classification for the fifth image. The correct classification in the fifth image is then used as an initial estimate for the sixth image where another satisfactory classification is obtained.

In contrast to this we look at the image sequence in Figure 5.8 which corresponds to the configuration of MEM where $\sigma_u = \sigma_v = 4$. In the first image of this sequence we already see that a building has been misclassified and remains misclassified in the second image. In the third image we see that the MEM converged to a set of

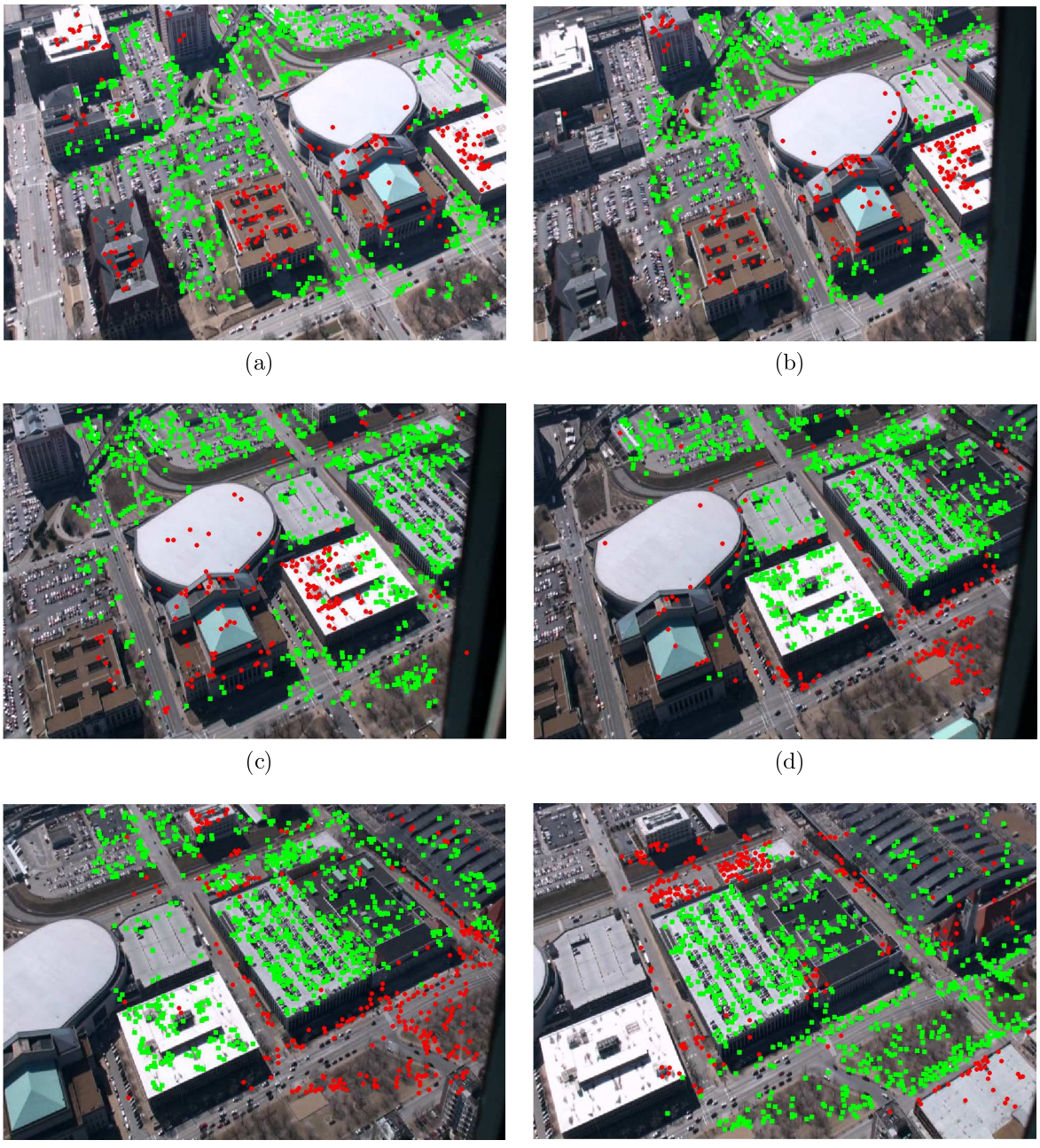


(a) (b)

(c) (d)

(e) (f)

Figure 5.7: SubSequence of Sequence 2. Configuration where $\sigma_u = \sigma_v = \sqrt{2}$



(a) (b)

(c) (d)

(e) (f)

Figure 5.8: SubSequence of Sequence 2. Configuration where $\sigma_x = \sigma_y = 4$

homography parameters that describe a plane that goes from the top of the parking garage in the right of the image to the ground plane in the left of the image. In the fourth image we see that the MEM has converged to the plane on the tops of the buildings like the JLinkage and RANSAC algorithms. It is here where we see the adverse effect of the initial guess. When it comes time to formulate an initial estimate for image five, our algorithm uses the classification from image four. Unfortunately, the classification was incorrect in image four, and the initial guess we create describes the wrong plane in the scene, that is the plane that spans the building top. The MEM then goes to refine this wrong estimate and once again converges to the tops of the buildings. This is also the case for image six. This is an example when the initial estimate can have a negative effect on the classification of the MEM and illustrates the concept discussed in Section5.4 which is that the quality of the classification depends on the initial estimate provided to the MEM.

5.3.2 The parameter Σ_x and its Effects

Something to be noted from Table 5.3 is that the configuration of MEM that perfomed the best in this experiment was the one where $\sigma_u = \sigma_v = \sqrt{2}$. However, we recall that the results obtained in Section 5.1 provided estimates of $\sigma_u = \sigma_v = 5$. It is in this section that the difference in these parameter values is discussed along with the effect that Σ_x has on the classification of MEM.

First, we refer to Section 5.2.1 where the effect of Σ_x was briefly described. In that section it was stated that Σ_x is used to calculate the likelihood for the point correspondences which determines how much the correspondence contributes to the M-Step of the algorithm. Low values for σ_u and σ_v tend to assign lower likelihood values while higher values tend to assign higher likelihood values. For instance, consider a correspondence whose calculated geometric error for a given set of homography parameters is 1 pixel in the u direction and 1 pixel in the v direction resulting in a total geometric error of 1.41 pixels. In many cases a point with such a small geometric error would be considered as a point that satisfies the homography constraint. Using $\sigma_u = \sigma_v = \sqrt{2}$, the likelihood for this correspondence would be 0.6065 while the likelihood calculated using $\sigma_u = \sigma_v = 5$ would be 0.9608.

From this example, it is clear that the values used to construct Σ_x significantly effect the likelihood of the correspondence. The question then becomes: "Under what circumstances would it be necessary to assign low likelihood values to all point correspondences?".

To answer this question, we refer back to the data set from this experiment. This data set was composed of airborne image sequences which contained point correspondences that were a significant distance from the camera. As previously mentioned the nature of these sequences emphasizes the importance of having a large baseline between the images since the perception of depth increases with the baseline. A similar relationship also exists between the geometric error of point correspondences and the baseline between the images where a larger baseline results in larger geometric errors for point correspondences that do not belong to the plane. This relationship becomes a problem in scenes such as the ones in Figure 5.7 where the geometric error for correspondences that do not belong to the ground plane are approximately 7 pixels due to the relatively small baseline compared to the distance of the points from the camera. This would result in $\sigma_u = \sigma_v = 5$ being a bad choice of parameters since a correspondence with such an error would receive a likelihood of roughly 0.35. One method to remedy this problem would be to increase the baseline between images which would increase the value of the geometric error of the non ground correspondences. However, as previously mentioned this would result in a smaller set of point correspondences collected between the two images since the area of overlap between the two images would most likely decrease. An alternative solution to this problem would be to adjust the value of σ_u and σ_v. Adjusting σ_u and σ_v to be smaller values would result in correspondences with a geometric error of 7 pixels to have a smaller likelihood, while still allowing the ground correspondences to have a significant effect when updating the parameters in the M-Step.

Overall σ_u and σ_v can be viewed as parameters whose values can be adjusted for different scenes that the MEM algorithm is applied. We restate that the experiment from Section 5.1 was not intended to find exact values of σ_u and σ_v but instead to confirm the assumption that the correspondence error observed in a set of matched correspondences followed a normal distribution. This explains the results observed in

Table 5.3, where a configuration of $\sigma_u = \sigma_v = \sqrt{2}$ outperformed other configurations of the MEM algorithm for this data set.

5.4 Multiple Planes - Sensitivity Analysis

The previous experiment provided an example where the initial estimate that was provided to the MEM algorithm caused the classification of the MEM to be incorrect. That example illustrates an important relationship that exists between the initial estimate provided to the MEM algorithm and the quality of the classification provided by the MEM algorithm. The general assumption that holds true even for the EM algorithm is that a "good" initial estimate should result in a "good" classification while a "bad" initial estimate has the potential to result in a "bad" classification. In terms of a "good" estimate, there is none better than the estimate which is the correct model parameters that are being solved for. In this situation, the algorithm should not stray far from this solution because it is the maximum of the complete data likelihood. On the other hand, a "bad" initial estimate is one that provides no distinction between classes when calculating the posterior probabilities in the E-Step. That is the posterior probabilities for all points for all classes are equivalent. The effect of a bad estimate is especially present when MEM is being used in a situation where outliers are accounted for by a nil class. In this scenario, a "bad" estimate would result in all of the correspondences being assigned to the nil class which would maximize the complete data likelihood making an accurate classification impossible.

Provided with this concept of good and bad estimates, two question arise. The first is "How can one evaluate the performance of an algorithm whose results depend on its initial estimate?". In the previous sections, the evaluation of the application of MEM to a specific problem was trivial because the calculation of the initial guess was embedded in the algorithm. However, in this section MEM is being applied to scenes with multiple planes and does not have any prior knowledge of the scene to construct an initial estimate. To present a set of results obtained by providing an estimate close to the desired solution would misrepresent the effectiveness of the

algorithm because this would most likely result in outstanding classifications. The same can be said about presenting a set of results obtained by providing horrible initial estimates which would downplay the effectiveness of the MEM algorithm.

In addition to this, the second question that arises is "Is there a way to quantify what a good and bad estimate is?". The previous discussion of good and bad estimates provides a general concept when it comes to the relationship between initial estimates and the resulting classification. However, what we would like to know is what a bad estimate is since its converse is how good the estimate is. To answer both of these questions, in this section an investigation is carried out that analysis how the algorithm performs with a range of initial estimates in a multiplane setting. Specifically, we will be performing a sensitivity analysis on the MEM algorithm for the parameters of φ_n, ϕ_n, d which describe the normal vector and distance value that compose the homography. This sensitivity analysis will provide us with a means to fairly evaluate the MEM algorithm and allow us to quantify a bad estimate.

The set of images used for this sensitivity analysis were obtained from the University of Oxford Visual Geometry Group's database. All of the images were of outdoor scenes that contained building faces which are the planar surfaces that the MEM needs to classify. In the sensitivity analysis a sweep was performed for each parameter while the other parameters were held constant. For the angular parameters φ_n and ϕ_n, the sweep consisted of adding an angular offset to the ground truth value of the parameter. This offset value ranged from -50 degrees to 50 degrees at a resolution of 1 degree. For the distance value, d, an offset value that ranged from -10 to 10 at a resolution of 0.1 was added to the ground truth value.

This sensitivity analysis was performed for three variations of the MEM algorithm: 1.) MEM that uses an optimization for the M-Step, 2.) MEM that determines the parameters $r, p, y, \varphi_t, \phi_t$ before classification and fixes these parameters for each class for every iteration of the algorithm and 3.) MEM that uses the Kalman Filter based M-Step. Each version of MEM was ran for different values of Σ_x and the quality of the classification was recorded. To measure the quality of the classification for each point in the sensitivity analysis, the Rand Index was calculated between the ground truth and the output of the MEM. The naming convention in the titles

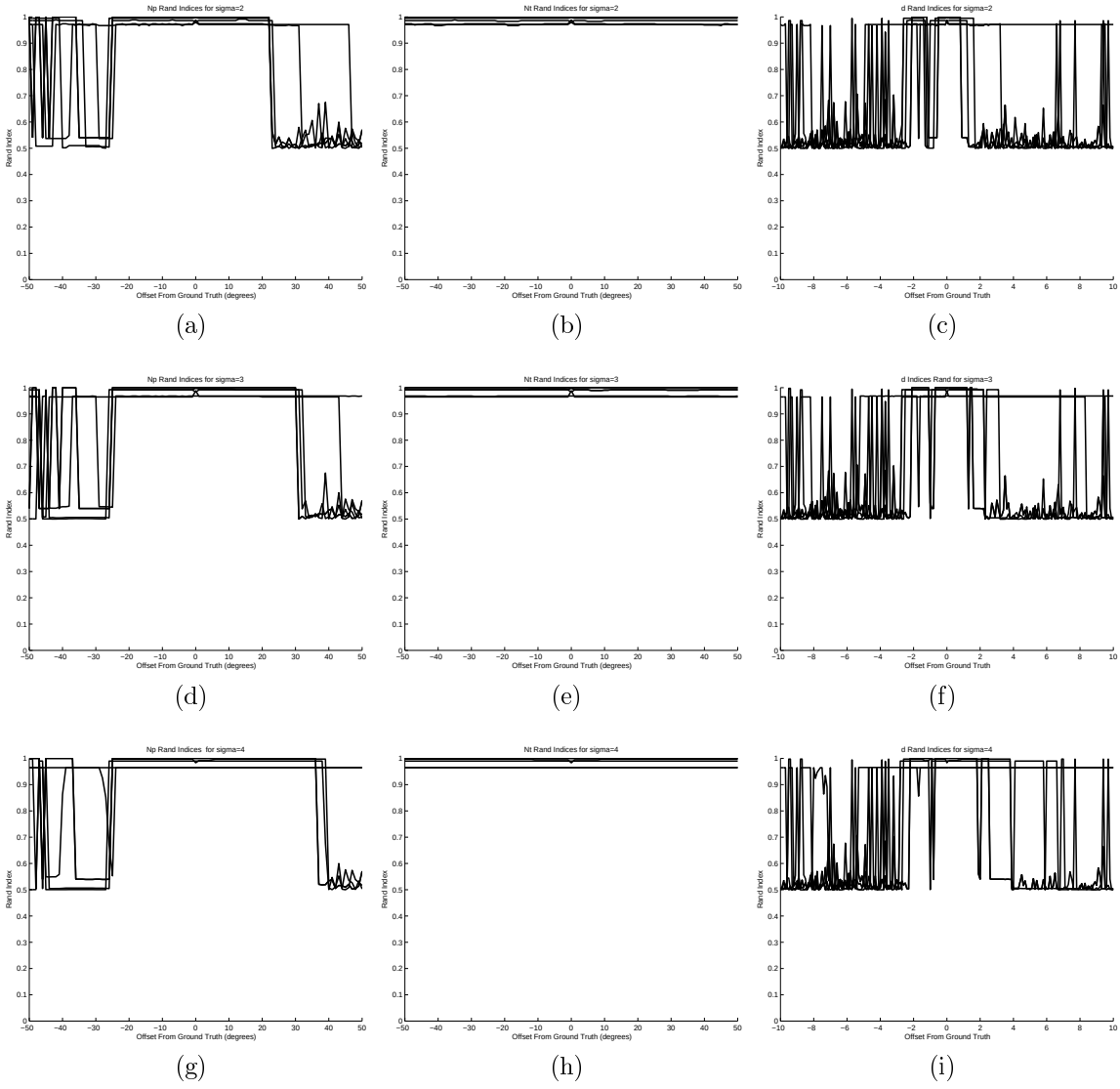


Figure 5.9: Sensitivity analysis results for optimization based MEM that solves for only the normal vector parameters and distance parameters

follow the following mapping: Np $=\varphi_n$, Nt $= \phi_n$, d $= d$.

The first variation we present sensitivity results for is the version of MEM which learns the motion parameters of the scene prior to estimating φ_n, ϕ_n, and d. By doing so, the number of parameters that need to be estimated using MEM is reduced to three parameters per plane. We restate that this is a valid approach to planar detection because many times the scene will undergo a single motion between cameras. That is, there are no moving objects in the scene and the only motion observed between images is the motion that the camera has undergone. The plots in Figure 5.9 show the three sensitivity analysis for these parameters.

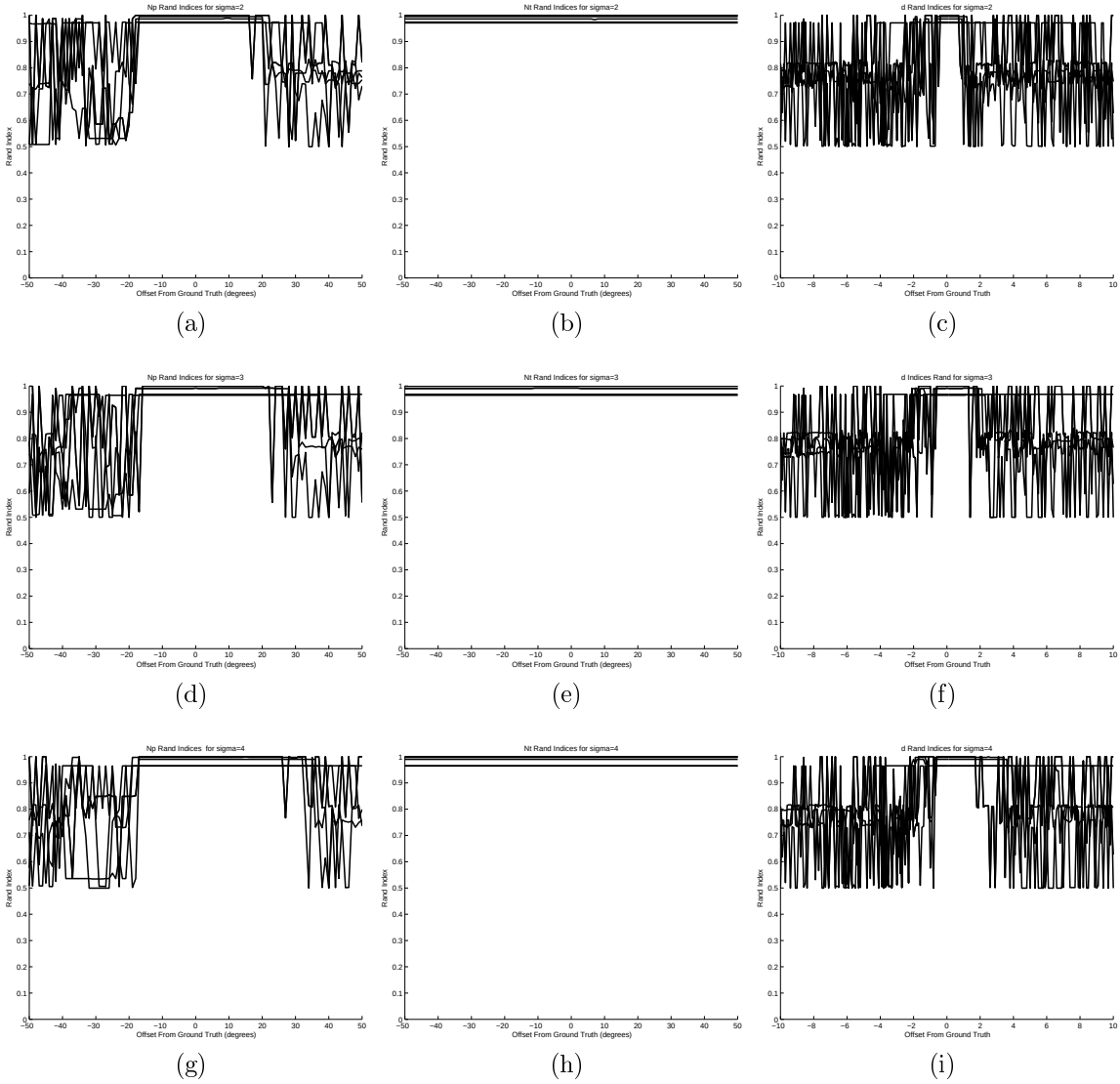
The next version of MEM that we provide sensitivity results for is the version of MEM which estimates a set of motion parameters along with the three parameters for each plane. While this results in five extra parameters being estimated by the MEM, only results for φ_n, ϕ_n, and d are presented for comparison with the other MEM variants. The results for this sensitivity analysis are provided in Figure 5.10.

The last version of MEM that we provide a sensitivity analysis for is that which employs Kalman Filtering in its M-Step. This version of MEM learns the motion parameters of the scene before estimating φ_n, ϕ_n, and d. The results for this sensitivity analysis are provided in Figure 5.11.

5.4.1 Comparison of Variations

The first observation to make from Figures 5.9, 5.10 and 5.11 is that all of the implementations achieved high Rand indices when provided with the ground truth of the parameters as an initial estimate. This was mentioned in the beginning of the section when it was stated that the best initial estimate that the MEM can be provided is the correct set of parameters. In this situation the figures show that the MEM algorithm does not diverge far from this solution since it is the maximum of the complete data likelihood. It is noted that some of the implementations in the figures do not achieve a Rand Index of exactly 1 when provided the ground truth. This comes from the fact that many of the scenes used in this sensitivity analysis contain planes that intersect each other. Many times there will exist misclassifications near these intersections because the correspondences really belong to both planes equally and must be assigned to one in the ground truth.

Further investigation of the figures shows that each implementation has a range of offsets that can be applied to the ground truth value of the parameters in which the MEM can provide an accurate classification. For all of the implementations, the range of values for ϕ_n is large, while the range for φ_n and d are more defined. The values that exist inside these ranges for each parameter are what we can consider "good" initial estimates for the parameter. Outside of these ranges, it can be seen that the graphs oscillate wildly and the MEM is occasionally able to achieve an accurate classification. It is noted that it is possible for the MEM to estimate a set

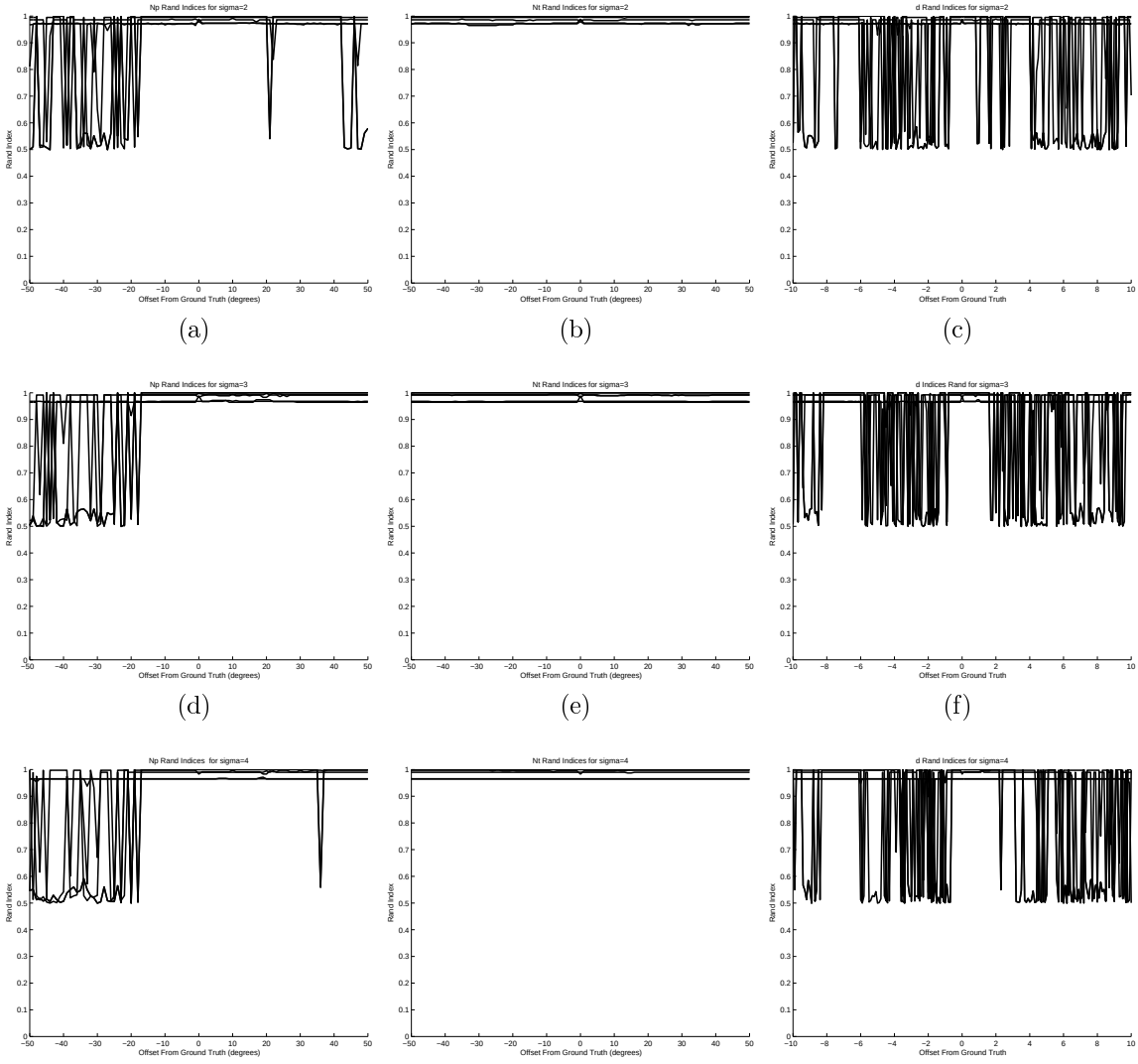


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 5.10: Sensitivity analysis results optimization based MEM that solves for global motion parameters in addition to normal vector parameters and distance parameters



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 5.11: Sensitivity analysis results for Kalman Filter based MEM

of parameters that lead to a correct classification when the initial estimate is outside of the range of "good" values for the parameter. Just because an initial estimate is outside of this range does not mean that it cannot be recovered from. Instead, values outside of the ranges should be viewed as values that have a higher risk of not being recovered from.

One relationship that is evident for all three implementations is the one that exists between the values used to construct Σ_x and the size of the range of "good" estimates. From all three implementation it can be seen that the size of the range increases as the values used to construct Σ_x increases. After the explanation in Section 5.3.2 it is apparent why this relationship exists. When the initial estimate gets further away from the ground truth, the likelihoods for all the correspondences decreases. However, by increasing the values used to construct Σ_x these likelihood values increase to the point where the offset can grow larger while still assigning the same likelihoods.

When comparing the various MEM implementations to one another, it can be seen that some implementations are more robust to the initial estimate than others. Specifically, the figures indicate that the implementation of MEM which solves for the motion parameters is the least robust implementation of the three presented. The reason for this comes from the extra motion parameters that must be estimated by the MEM. By estimating the motion parameters in the MEM algorithm, we are increasing the degrees of freedom that the MEM must update. That is, even though only one parameter may be different from the ground truth, the M-Step of the MEM algorithm will update the three parameters that describe the plane along with the motion parameters. This can cause the motion parameters to diverge from their true values which will cause the MEM to fall into a local minimum. This problem would be amplified by an implementation which solves a set of motion parameters for each homography.

This experiment demonstrated that there does exist a range of values that the initial estimate can be for a parameter where the MEM can still return an accurate classification. It was shown that these ranges grow with Σ_x and that the implementations that solve for less parameters are more robust to the initial estimate. Some

Figure 5.12: Example Classifications of the MEM algorithm for a multiple plane setting

results of the classifications listed in the figures are provided in Figure 5.12.

Chapter 6

Future Work

Future work of this topic consists of the following three areas. First the investigation of optimization algorithms to use for the modified maximization step was limited in this work to the Nelder-Mead Simplex algorithm. Other optimizations such as Levenberg-Marquardt or genetic algorithms such as Particle Swarm might prove to be useful substitutions for the Simplex algorithm. Advantages might exist for these other algorithms that can increase the performance of the MEM algorithm's estimation.

The second area of investigation is to define methods that can be used for initialization of the algorithm. The importance of the initial estimate provided to the MEM algorithm was discussed throughout this work with the final observation being that initial estimate directly effects the estimation of the MEM algorithm. Developing methods to initialize the MEM algorithm are important to be able to keep the algorithm competitive with current state of the art methods. This is especially the case when using MEM in a setting with multiple planar regions.

The last area of investigation also contributes to the algorithm's competitiveness with current state of the art algorithms which is to take outliers into account for a multiple plane setting. In this work, all of the point correspondences in a multiple plane setting were assumed to belong to one of the planes in the scene. In the presence of outliers, the current version of the MEM algorithm would be required to label the outliers to one of the classes. This would result in the outliers having the opportunity to effect the update of the estimates. In this work, outliers were

accounted for when the MEM algorithm was used to estimate the parameters of a single plane in the scene. However, the extension of this method to a multiple plane setting is questionable and requires further attention.

Chapter 7

Conclusion

This work presented the Modified Expectation Maximization (MEM) algorithm that can be used for estimating the model parameters that define the homographies that are induced by planar regions in a planar segmentation task. This algorithm differs from the Expectation Maximization (EM) algorithm in that it does not use a closed form solution for its maximization step. This difference is necessary in a planar segmentation task where the closed form solution for the maximization step is impractical to derive. Two implementations were presented for the MEM algorithm's modified maximization step: one that uses an optimization and another that uses a Kalman Filter. Both implementations are used to update the current estimate of the model parameters which is the original intent of the EM algorithms maximization step.

In addition to the MEM algorithm, this work also presented a Kalman Filter based method that can be used to refine an estimate of the motion parameters that describe the transformation between two cameras. In order to derive the necessary Kalman Filter equations for this method, a 3D reconstruction algorithm was presented that can be used to calculate a 3D point in space from a point correspondence and transformation using the rays of the correspondence. This 3D reconstruction algorithm is essential to the derivation of the Kalman Filter equations because it can be used to define a simple constraint equation. This method was provided as a way for the MEM algorithm to reduce the number of parameters it needs to estimate by solving for the motion parameters before doing the MEM iterations.

Experiments show that variations of the MEM algorithm are effective in both single plane and multiple plane settings. In the mobile robot follower experiment the MEM algorithm was employed to detect the ground plane of a scene. The two tested versions of the MEM algorithm were able to achieve competitive results with current state of the art methods such as RANSAC and JLinkage in this task. The effectiveness of the MEM algorithm in a plane tracking scenario was also provided for an aerial sequence. This experiment demonstrated the advantage MEM has over other methods by being able to incorporate prior information about the location of the plane from previous image pairs. However, this experiment also demonstrated the importance of the initial estimate provided to the MEM algorithm and its effects on the MEM's converged estimates. Finally, a sensitivity analysis was performed on the MEM algorithm in a multiple plane scenario to illustrate how the MEM algorithm converges for different initializations. This analysis was successful in showing that there is a range of estimates that can be used from which the MEM algorithm can achieve an accurate estimate. Overall, this work shows that the MEM algorithm has the potential to be a competitive method for planar segmentation in image pairs.

Bibliography

[1] K. min Han and G. DeSouza, "Instantaneous geo-location of multiple targets from monocular airborne video," in *IEEE International Geoscience and Remote Sensing Symposium*, 2009.

[2] S. Lenser and M. Veloso, "Visual sonar: fast obstacle avoidance using monocular vision," in *IEEE International Conference on Intelligent Robots and Systems*, 2003.

[3] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Image Understanding Workshop*, 1981.

[4] N. Ohnishia and A. Imiya, "Dominant plane detection from optical flow for robot navigation," *Pattern Recognition Letters*, 2006.

[5] Y. geun Kim and H. Kim, "Layered ground floor detetion for vision-based mobile robot navigation," in *IEEE International Conference on Robotics and Automation*, 2004.

[6] M. M. V. H. Nikolay Chumerin, "Ground plane estimation based on dense stereo disparity," in *The Fifth International conference on Neural Networks and Artificial Intelligence*, 2008.

[7] J.-P. T. Raphael Labayrade, Didier Aubert, "Real time obstacle detection in stereovision on non flat road geometry through v-disparity representation," in *IEEE Intelligent Vehicle Symposium*, 2002.

[8] P. Lombardi, M. Zanin, and S. Messelodi, "Unified stereovision for ground, road, and obstacle detection," in *Intelligent Vehicles Symposium*, 2005.

[9] J. Zhao, J. Katupitiya, and J. Ward, "Global correlation based ground plane estimation using v-disparity image," in *IEEE International Conference on Robotics and Automation*, 2007.

[10] A. Cherian, V. Morellas, and N. Papanikolopoulos, "Accurate 3d ground plane estimation from a single image," in *IEEE International Conference on Robotics and Automation*, 2009.

[11] P. Fornland and C. Schnorr, "A robust and convergent iterative approach for determining the dominat plane from two views without correspondence and calibration," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997.

[12] D. Lowe, "Object recognition from local scale-invariant features," in *IEEE International Conference on Computer Vision*, 1999.

[13] J. Zhou and B. Li, "Homography-based ground detection for a mobile robot platform using a single camera," in *IEEE International Conference on Robotics and Automation*, 2006.

[14] M. Lourakis and S. Orphanoudakis, "Visual detection of obstacles assuming a locally planar ground," in *3rd ACCV*, 1998.

[15] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.

[16] F. Mufti, R. Mahony, and J. Heinzmann, "Spatio-temporal ransac for robust estimation of ground plane in video range images for automotive applications," in *IEEE Conference on Intelligent Transportation Systems*, 2008.

[17] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision Second Edition*. Cambridge University Press, 2004.

[18] N. Pears and B. Liang, "Ground plane segmentation for mobile robot visual navigation," in *IEEE International Conference on Intelligent Robots and Systems*, 2001.

[19] D. Sinclair and A. Blake, "Quantitative planar region detection," *International Journal of Computer Vision*, 1996.

[20] K. Aires, H. Araujo, and A. Medeiros, "Plane detection using affine homography," in *Visualization, Imaging and Image Processing*, 2008.

[21] J. Piazzi and D. Prattichizzo, "Plane detection with stereo images," in *IEEE International Conference on Robotics and Automation*, 2006.

[22] Y. Kanazawa and H. Kawakami, "Detection of planar regions with uncalibrated stereo using distributions of feature points," in *British Machine Vision Conference*, 2004.

[23] E. Vincent and R. Laganiere, "Detecting planar homographies in an image pair," in *2nd International Symposium on Image and Signal Processing and Analysis*, 2001.

[24] M. Zuliani, C. Kenney, and B. Manjunath, "The multiransac algorithm and its application to dtect planar homographies," in *IEEE International Conference on Image Processing*, 2005.

[25] R. Toldo and A. Fusiello, "Robust multiple structures estimation with j-linkage," in *European Conference on Computer Vision*, 2008.

[26] A. Y. Yang, S. Rao, A. Wagner, and Y. Ma, "Segmentation of a piece-wise planar scene from perspective images," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[27] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[28] C. V. J. Anubhav Agarwal and P. J. Narayanan, "A survey of planar homography estimation techniqes," 2005.

[29] O. D. Faugeras and F. Lustman, "Motion and structure from motion in a piecewise planar environment," *International Journal of Pattern Recognition and Artificial Intelligence*, 1988.

[30] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society*, 1977.

[31] J. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, 1965.

[32] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, 1960.