CONSTRAINED SELF-CALIBRATION

JEFFREY IRA MENDELSOHN

A DISSERTATION

in

COMPUTER AND INFORMATION SCIENCE

Presented to the Faculties of the University of Pennsylvania in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy.

Bajcsy

Dr. Ruzena Bajcsy

Supervisor of Dissertation

Jean H Gallier

Dr. Jean Gallier

Graduate Group Chairperson

Acknowledgements

I thank Mr. Felker for his assistance and advice during my studies and especially for helping me find a way to fund my first few semesters. Special thanks to Dr. Dennis DeTurck for his patronage throughout my graduate studies and Dr. Max Mintz for the support that encouraged me to complete the doctoral program.

My gratitude to Dr. Eero Simoncelli and Dr. Kostas Daniilidis for their mentoring and advice both within and without of academia and Dr. Ruzena Bajcsy for her guidance, support, and service as my advisor.

Finally, I thank my parents Deanna and Stanley Mendelsohn and sister Meryl for their encouragement, advice, and support in all my endeavors.

ABSTRACT

CONSTRAINED SELF-CALIBRATION

JEFFREY IRA MENDELSOHN
RUZENA BAJCSY

This dissertation focuses on the estimation of the intrinsic camera parameters and the trajectory of the camera from an image sequence. Intrinsic camera calibration and pose estimation are the prerequisites for many applications involving navigation tasks, scene reconstruction, and merging of virtual and real environments. Proposed and evaluated is a technical solution to decrease the sensitivity of self-calibration by placing easily identifiable targets of known shape in the environment. The relative position of the targets need not be known *a priori*. Assuming an appropriate ratio of size to distance these targets resolve known ambiguities. Constraints on the target placement and the cameras' motions are explored. This dissertation also includes improvements in pose estimation and a novel algorithm for weak-calibration.

Contents

List of Tables

List of Figures

Preface

This dissertation focuses on the estimation of the intrinsic camera parameters and the trajectory of the camera from an image sequence. Intrinsic camera calibration and pose estimation are the prerequisites for many applications involving navigation tasks, scene reconstruction, and merging of virtual and real environments.

Conventional techniques for solving this problem include off-line camera calibration followed by pose estimation. This method calibrates the camera with regard to a known target and then estimates the position of the camera relative to the target in successive frames. The primary difficulties of this technique are inaccuracies of calibration from one frame, the impracticality of keeping a target in the camera's field of view, and the degeneration of pose estimation to an ill-conditioned problem as the object becomes small in the image. Recently, self-calibration algorithms have been developed. Self-calibration has the same goal of standard calibration but does not rely on the use of a known target. Algorithms use a set of correspondences - typical points or lines - over a set of images to estimate the camera's intrinsic parameters. These correspondences can also be used to estimate camera trajectory. Unfortunately, self-calibration is not reliable due to the confounding between intrinsic parameters and motion.

Proposed and evaluated is a technical solution to decrease the sensitivity of self-calibration by placing easily identifiable targets of known shape in the environment. The relative position of the targets need not be known *a priori*. Assuming an appropriate ratio of size to distance these targets resolve known ambiguities. Constraints on the target placement and the cameras' motions are explored.

The dissertation also includes improvements in pose estimation and a novel algorithm for weak-calibration.

Chapter 1

Introduction

1.1 Problem Definition and Motivation

This dissertation explores the problem of extracting camera calibration and trajectory information from a sequence of images. The most general statement of this problem is:

> Given a set of images and the knowledge of which camera produced each image, estimate the intrinsic calibration parameters (e.g. focal length, image center, and scale factor) of each camera as well as the position of each camera, relative to a world coordinate system, at recording time.

The primary difficulties of this problem stem from the confounding of the intrinsic and motion parameters by the projection process.

A key use of such a system is as a solution to the registration problem of augmented reality; the estimation of the alignment between the real and virtual world [Azu97]. Applications span many industries. In surgery CAT-reconstructions are superimposed on the real images of a patient [Mel95]. During maintenance of laser printers [FMS93], personnel are guided step-by-step by showing the appropriate motions through wire-frame animation of the parts to be exchanged or repaired. Azuma mentions the application of "Living History" where a tourist can walk near Acropolis in Athens seeing Acropolis in its original BC construction including the statues of that time [Azu97].

Other applications include baseline estimation for stereo vision and robot localization. Stereo is the computation of scene depths from a set of images where the relative orientations of the cameras as well as the cameras' calibration information are known (see [DA89] for a review). Localization refers to the task of determining the observer's position and orientation relative to a world coordinate system [Sig85, Dru87, Cox89].

1.2 Current Solutions

The problem of estimating the cameras' intrinsic parameters as well as the trajectories of the cameras can be solved in two stages. First the camera can be calibrated relative to a known target and then pose estimations relative to this target can be performed in successive frames. Camera calibration is the estimation of the camera's intrinsic parameters, such as the focal length and the center of the image, as well as its extrinsic parameter which are the rotation and translation from a known target (see [Tsa89, Fau93] for a review). Pose estimation is a process for determining the position of a calibrated camera to a known target [KH94, PHYT95, HDLC97].

The key difficulty with classic camera calibration algorithms is the reliance on only one frame of information. From one view point there is typically insufficient information to accurately separate the effects of the intrinsic and the extrinsic parameters [Bla97]. While the projection of the object will appear accurate in the calibration image, there will not exist a set of extrinsic parameters that when combined with the previously estimated intrinsic parameters can accurately project the object into images taken far from the calibration view point; implying the intrinsic parameters are inaccurate and unusable in a large workspace [Bla97].

Furthermore, the pose estimation problem is not practically solved when using one object in a large workspace. First, keeping the object in the cameras' fields-of-view may be impractical. Second, pose estimation becomes an ill-conditioned problem if the object becomes small in the image [KH94].

Other techniques for addressing the problem include structure-from-motion (for a review see [HN94]) and self-calibration [MF92]. Structure-from-motion estimates the camera's trajectory as well as the scene structure up to an ambiguity; if the camera's intrinsic calibration is known then the reconstruction is up to a similitude transformation and with unknown intrinsic calibration a reconstruction up to a projective transformation. Self-calibration is the estimation of camera calibration from a sequence of images not involving a calibration target. Generally, point or line features are found and tracked to use as input to the self-calibration procedure. While some self-calibration algorithms do not involve reconstructing the camera's trajectory or the scene structure, most do estimate these values [MF92, Har93]. Self-calibration approaches are known to suffer the problems of structure-from-motion as well as the confounding between intrinsic parameters and motion [Oli98].

1.3 Overview

Proposed in this dissertation is a technical solution to decrease the sensitivity of self-calibration by placing small, easily identifiable targets of known shape in the environment. Assuming an appropriate ratio of size to distance these targets resolve known ambiguities. The proposed algorithm can be applied in any moving camera application and enables the direct merging of several views into a single 3D-representation such as needed during a stereo reconstruction of a room. For other applications the path of the observer is recovered and can be used for global navigation.

The effects of commonly incorporated constraints - rigidly linked cameras, planar motion of the cameras, and coplanarity of targets - with regard to the reconstruction accuracy are explored. Qualitative results are provided by sets of augmented reality images and bird's-eye-views of the target and observer positions. The quantitative results compare estimated and measured values in three-space as opposed to image space; camera baseline distance, distances between targets, and coplanarity of coplanar targets.

Secondary items proposed in this dissertation are improvements to projective approximation pose estimation algorithms and a novel optical flow algorithm which constrains the estimated flow field to one producible by a rigid motion.

1.4 Notation

In equations matrices are denoted by bold capital letters (e.g. $\mathbf{M}$), vectors by bold lower-case letters (e.g. $\mathbf{q}$), and scalars by lower-case letters (e.g. α, z). Unit vectors are denoted with a hat (e.g. $\hat{\mathbf{z}}$). Rotation matrices are denoted with a subscripted R-matrix (e.g. $\mathbf{R}_f$), translation vectors by subscripted t-vectors (e.g. $\mathbf{t}_t$), points in Euclidean three-space by subscripted x-vectors (e.g. $\mathbf{x}_c$), and points in projective three-space by $\mathbf{p}$.

The unit vectors along each axis are denoted by $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$:

$$\hat{\mathbf{x}} \equiv \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \hat{\mathbf{y}} \equiv \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \hat{\mathbf{z}} \equiv \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} .$$

All subscripted vectors $\mathbf{x}_\diamond$ and the vector $\mathbf{p}$ are defined to have the following components:

$$\mathbf{x}_\diamond \equiv \begin{bmatrix} x_\diamond \\ y_\diamond \\ z_\diamond \end{bmatrix} \quad \mathbf{p} \equiv \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} .$$

Finally, the matrix $\mathbf{C}$ is used to denote the matrix of camera intrinsics:

$$\mathbf{C} \equiv \begin{bmatrix} fs & 0 & c_u \\ 0 & f & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

where f is the focal length, c_u and c_v provide the coordinates of the imager center, and s represents any scale factors such as aspect ratio of pixels and sampling rate discrepancies.

The results of the cross-product operation between two vectors $\mathbf{x}$ and $\mathbf{y}$ can be written as the product of a skew-symmetric matrix and a vector. This conversion to a skew-symmetric matrix from a vector will be denoted by:

$$\mathbf{x}^\times \equiv \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}$$

1.5 Frames of Reference

The world coordinate system is an arbitrarily selected reference frame. Typically, a useful system will be selected such as the position of a particular camera in the first frame of a sequence or the reference frame of some target. Points in the world coordinate system are denoted by $\mathbf{x}_w$.

Images are assumed to be acquired in a sequence; each frame has an index. Each frame's coordinate system is a rigid transformation of the world system and will be denoted by a rotation matrix $\mathbf{R}_f$ and a translation vector $\mathbf{t}_f$. The coordinates of the transformed world point $\mathbf{x}_w$ relative to the frame f is given by $\mathbf{x}_f = \mathbf{R}_f\mathbf{x}_w + \mathbf{t}_f$.

A camera's frame of reference will always be a rigid transformation of the appropriate image frame's coordinate system; $\mathbf{x}_c = \mathbf{R}_s\mathbf{x}_f + \mathbf{t}_s$. If there are no rigidly linked cameras these $\mathbf{R}_s$ are identity matrices and $\mathbf{t}_s$ are zero vectors. In case of a stereo camera rig, one camera is chosen as the base camera and it has no rotation or translation. The other cameras use $\mathbf{R}_s$ and $\mathbf{t}_s$ to covert the frame coordinates into camera coordinates. Finally, the coordinate system of a camera is defined to have the focal point as the origin, the optical axis as the z-axis, and the x-axis is aligned with the x-axis of the CCD chip (from the upper-left corner to the upper-right corner of the chip).

Combining the rigid coordinate mappings from world-to-frame and from frame-to-camera yields the world-to-camera transformations:

$$\mathbf{x}_c = \mathbf{R}_s(\mathbf{R}_f\mathbf{x}_w + \mathbf{t}_f) + \mathbf{t}_s \ . \tag{1.1}$$

When using targets, every target's features are defined with respect to that target's reference frame. A point $\mathbf{x}_t$ in the target's coordinate system is mapped into the world coordinate system by a rotation $\mathbf{R}_t$ and a translation $\mathbf{t}_t$ resulting in the transformation equation $\mathbf{x}_w = \mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t$.

Combining the target-to-world coordinate mapping with the world-to-camera mapping given in Equation 1.1 yields:

$$\mathbf{x}_c = \mathbf{R}_s[\mathbf{R}_f(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t) + \mathbf{t}_f] + \mathbf{t}_s \ . \tag{1.2}$$

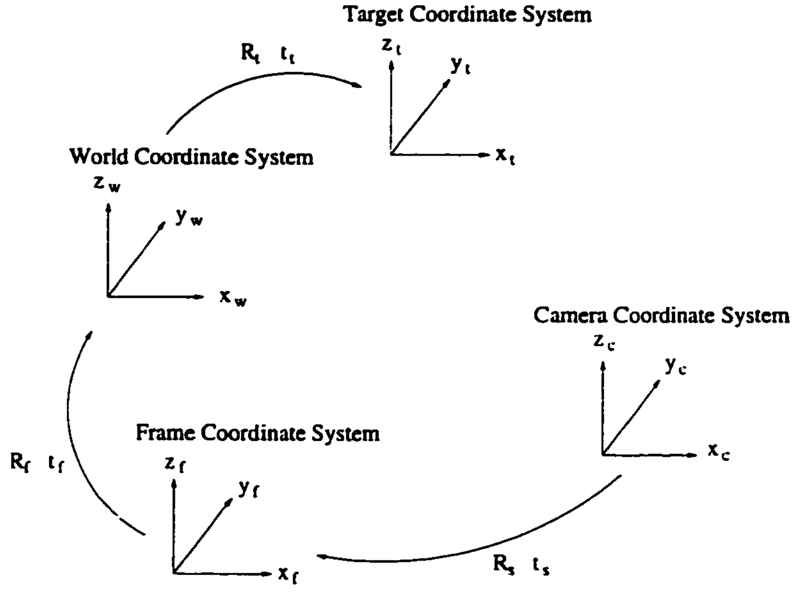


Figure 1.1: Coordinate systems.

1.6 Projection Models

A projection model defines the mapping between a spatial point in the camera's frame of reference $\mathbf{x}_c$ and its projection in the image $\mathbf{p}$. In reality, this is a very complex process involving ray tracing through a set of lenses. In practice, the projection of a point is well approximated by the perspective - or pinhole - projection model and is a common model in computer vision literature.

While a significant simplification, the perspective projection model is non-linear. Two common simplifications of perspective projection are paraperspective projection and weak-perspective projection.

1.6.1 Perspective Projection Model

Perspective projection is a two part process. First the spatial point is projected to the image plane and then an affine transformation is applied to represent the effects of the intrinsic camera parameters. Under perspective projection, the projection of a spatial point is the intersection of the image plane and the line connecting the spatial point to the focal point (Figure 1.2). Mathematically, perspective projection is provided by the following two equations:

$$u = fs\frac{\hat{\mathbf{x}}^T\mathbf{x}_c}{\hat{\mathbf{z}}^T\mathbf{x}_c} + c_u \qquad v = f\frac{\hat{\mathbf{y}}^T\mathbf{x}_c}{\hat{\mathbf{z}}^T\mathbf{x}_c} + c_v$$

where f is the focal length measured in pixels, c_u and c_v are the coordinates of the image center measured in pixels, and s is a scale factor accounting for the pixel aspect ratio and sampling rate discrepancies. These equations are expressed in vector notation as:

$$\mathbf{p} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} fs & 0 & c_u \\ 0 & f & c_v \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_c\frac{1}{\hat{\mathbf{z}}^T\mathbf{x}_c} = \mathbf{C}\mathbf{x}_c\frac{1}{\hat{\mathbf{z}}^T\mathbf{x}_c} . \qquad (1.3)$$

Note that a camera is a physical device and there are limits with regards to which points can be imaged; a camera can not image a point behind it. The field-of-view of a camera will be used to express the portion of space that can be imaged by the camera.

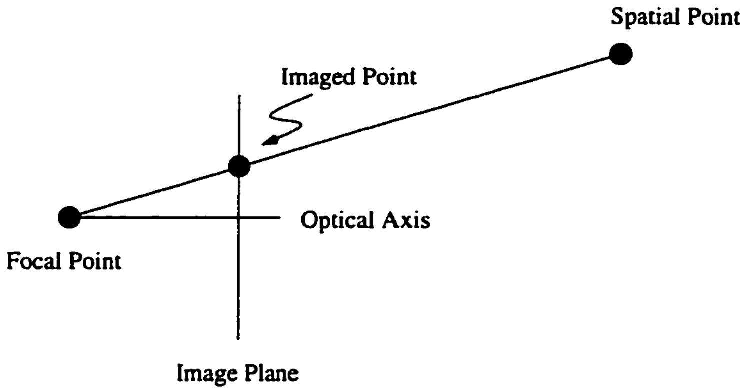


Figure 1.2: Perspective projection of a point.

1.6.2 Paraperspective Projection Model

Perspective projection is an approximation of the true projection process involving lenses and perhaps mirrors. Besides the accuracy of this approximation, there is little reason to prefer the perspective projection model over any other model; it might be advantageous to have a simplier model if the new approximation is sufficient for a variety of tasks. One such useful model is paraperspective projection [Alo90].

Under paraperspective projection a reference point $\mathbf{x}_o$ - typically the centroid of a group of points or the translation vector for an object - is chosen. The reference plane is defined to be the plane parallel to the image plane and through the reference point. A spatial point is projected into the image by two projections and the affine transformation that incorporates the camera's intrinsic parameters. The first projection is of the spatial point onto the reference plane. This projected point is then mapped into the image plane using perspective projection. The projection onto the reference plane is performed by finding the intersection of the reference plane with the line through the spatial point that is parallel to the line connecting the focal point and the reference point (Figure 1.3).

The first projection onto the reference plane is represented mathematically as:

$$\mathbf{x}_r = \mathbf{x}_c - \mathbf{x}_o \frac{\hat{\mathbf{z}}^T(\mathbf{x}_c - \mathbf{x}_o)}{\hat{\mathbf{z}}^T \mathbf{x}_o} = \left(\mathbf{I} - \mathbf{x}_o \hat{\mathbf{z}}^T \frac{1}{\hat{\mathbf{z}}^T \mathbf{x}_o}\right) \mathbf{x}_c + \mathbf{x}_o \ .$$

Combining with the second projection modified by the intrinsic camera parameters:

$$\mathbf{p}^p = \mathbf{C}\mathbf{x}_r \frac{1}{\hat{\mathbf{z}}^T \mathbf{x}_r} = \mathbf{C}\left[\left(\mathbf{I} - \mathbf{x}_o \hat{\mathbf{z}}^T \frac{1}{\hat{\mathbf{z}}^T \mathbf{x}_o}\right) \mathbf{x}_c + \mathbf{x}_o\right] \frac{1}{\hat{\mathbf{z}}^T \mathbf{x}_o}$$

where $\mathbf{p}^p$ denotes the projection of a point under paraperspective projection. This simplicity of this projection model is more apparent when the components of the matrices and vectors are written explicitly:

$$\mathbf{p}^p = \mathbf{C}\left(\begin{bmatrix} 1 & 0 & -\frac{x_o}{z_o} \\ 0 & 1 & -\frac{y_o}{z_o} \\ 0 & 0 & 0 \end{bmatrix} \mathbf{x}_c \frac{1}{z_o} + \begin{bmatrix} \frac{x_o}{z_o} \\ \frac{y_o}{z_o} \\ 1 \end{bmatrix}\right) \ . \tag{1.4}$$

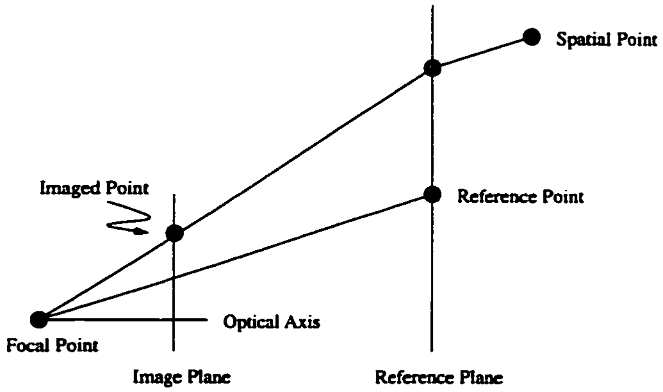


Figure 1.3: Paraperspective projection of a point.

1.6.3 Weak-Perspective Projection Model

A further simplification of the paraperspective projection model is requiring the reference point to be on the optical axis [Alo90]. The weak-perspective model is hence given by:

$$\mathbf{p}^w = \mathbf{C}\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{x}_c \frac{1}{z_o} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right) . \tag{1.5}$$

where $\mathbf{p}^w$ denotes the projection of a point under weak-perspective projection. Figure 1.4 depicts the projection of a spatial point under weak-perspective projection.

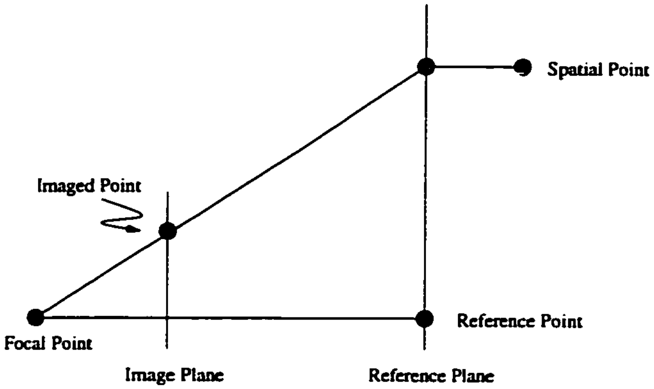


Figure 1.4: Weak-perspective projection of a point.

Chapter 2

Calibration and Pose

2.1 Overview

The problem of estimating the cameras' intrinsic parameters as well as the trajectory of the cameras can be solved in two stages. First the camera can be calibrated relative to a known target and then pose estimations relative to this target can be performed in successive frames.

Camera calibration is the estimation of the camera's intrinsic parameters, such as the focal length and the center of the image, as well as its extrinsic parameter which are the rotation and translation from a known target (Figure 2.1). Assuming the target defines the world coordinate system, the mapping from a point in the world system to its projection is defined by combining the mapping from the world coordinate system to the camera's coordinate system, given by Equation 1.1, with the perspective projection model described by Equation 1.3:

$$\mathbf{p} = \mathbf{C}[\mathbf{R}_s(\mathbf{R}_f\mathbf{x}_w + \mathbf{t}_f) + \mathbf{t}_s]\frac{1}{\hat{\mathbf{z}}^T(\mathbf{R}_s(\mathbf{R}_f\mathbf{x}_w + \mathbf{t}_f) + \mathbf{t}_s)} \; .$$

Assuming only one camera is being calibrated, $\mathbf{R}_s$ is an identity matrix and $\mathbf{t}_s$ a zero vector:

$$\mathbf{p} = \mathbf{C}(\mathbf{R}_f\mathbf{x}_w + \mathbf{t}_f)\frac{1}{\hat{\mathbf{z}}^T(\mathbf{R}_f\mathbf{x}_w + \mathbf{t}_f)} \; .$$

Camera calibration is the estimation of the calibration matrix $\mathbf{C}$, rotation matrix $\mathbf{R}_f$, and translation vector $\mathbf{t}_f$ from the knowledge of a target $\mathbf{x}_w$ and the observations $\mathbf{p}$. A

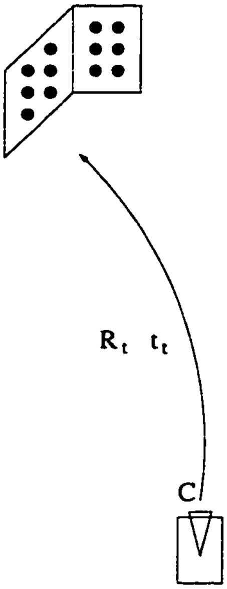


Figure 2.1: Depiction of the unknowns in camera calibration. They are the intrinsic parameters C, the rotation $\mathbf{R}_t$, and translation $\mathbf{t}_t$.

commonly used algorithm to solve this problem is [Tsa86]. Other methods and analysis of calibration techniques are available in [Fau93, LT88, Tsa89, WS94].

The problem of object pose estimation will be defined as finding the rigid transformation from the object frame to the camera frame given the camera projection model and calibration information, a set of points described in an object frame, and the projection of these points (Figure 2.2).

Object pose estimation has many uses in computer vision: object positioning, docking (moving the camera to a set transformation from the target), and cartography. The key difficulties in solving the problem stem from the constraints of a rotation matrix and having only the projection of the object points for data. Furthermore the data is not perfect. For the object pose problem, a primary concern is the localization error; a measure of how well the data points correspond to the true projection of the object points. Sources of localization error include the pixelization of the image by the camera, blur caused by the feature not being in focus, and incorrect pixel values from defects in the CCD [HK94].

Previous approaches to solving the object pose problem can be classified into two broad categories: closed-form solutions and numerical solutions. Closed-form solutions make use of a finite number of correspondences and solve the object pose problem by directly solving

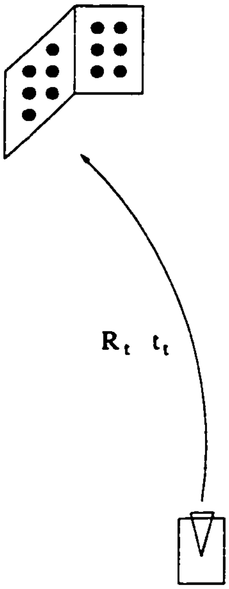


Figure 2.2: Depiction of the unknowns in pose estimation. They are the rotation $\mathbf{R}_t$ and translation $\mathbf{t}_t$ of the target.

for the transformation parameters in the set of projection equations. Such solutions exist for three points [FB81], four coplanar points [HYH85], and four points in general position [HN91, HCLL89]. While it is possible to derive closed-form solutions to overconstrained pose estimation problems, it is exceedingly difficult since the equations involve non-linear constraints. This is addressed by the numerical solutions.

Ganapathy [Gan84] proposed a linear solution on the assumption that the constraints on the rotation matrix need not be imposed; the rotation matrix is given nine degrees of freedom. This algorithm is extremely susceptible to noise mainly because the orthogonality constraints are ignored [PHYT95]. Numerical methods that correctly constrain the problem [HCLL89, Low87, Yua89] require a good initial estimate of the transformation parameters. This is a major limitation created, primarily, by the minimization technique. A state-of-the-art pose estimator by Phong et al. [PHYT95] uses a trust-region minimization technique which provides an excellent convergence rate and, essentially, removes the need for an initial estimate.

A relatively new sub-class of the numerical solutions can be defined as solving the pose estimation problem under an approximation to the desired projection model [DD95, HDLC97]. For instance, using weak-perspective or paraperspective projection as approximations to perspective projection. Clearly, it is unlikely for an approximation method to provide as accurate results as a method based on the true projection model. This is mostly overcome by iteratively moving the image points towards those that would have been produced by the approximation model's projection equations. While these algorithms do not converge for very close objects, in practice this is not a concern. Furthermore, these algorithms do not enforce the constraints of the transformation's rotation matrix which can make the algorithms unreliable in the presense of localization error [PHYT95]. In total, the execution time required for pose estimation is dramatically reduced at the expense of accuracy.

Simple techniques for improving the projection approximation algorithms are presented. The new algorithm substantially improves the estimation of object orientation. Also, at short distances the error in estimated position is dramatically decreased. As a result of these improvements, the presented algorithm provides a pose estimates useful for many applications; the algorithm is fast and accurate over a much larger set of distances. For comparison, the techniques presented in [PHYT95], [DD95], and [HDLC97] are reviewed in detail.

2.1.1 Phong, Horaud, Yassine, and Tao 1995

This paper provides an interesting statement which guides its development:

> Since the object pose from a single view problem is nonlinear, choices for (i) the mathematical representation of the problem, (ii) the error function to be minimized, and for (iii) the optimization method are crucial.

The perspective projection equation (Equation 1.3):

$$\mathbf{p} = \mathbf{C}\mathbf{x}_c \frac{1}{\hat{\mathbf{z}}^T \mathbf{x}_c}$$

is combined with the target-to-camera mapping given by Equation 1.2 assuming the camera's coordinate system is the world coordinate system:

$$\mathbf{p} = \mathbf{C}(\mathbf{R}_t \mathbf{x}_t + \mathbf{t}_t) \frac{1}{\hat{\mathbf{z}}^T (\mathbf{R}_t \mathbf{x}_t + \mathbf{t}_t)} \ .$$

First, since the intrinsic calibration parameters $\mathbf{C}$ are known, the equation is rewritten to combine the calibration with the measurements:

$$\mathbf{C}^{-1}\mathbf{p} = (\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t)\frac{1}{\hat{\mathbf{z}}^T(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t)} \ .$$

Second, the effect of the depth term $\frac{1}{\hat{\mathbf{z}}^T(\mathbf{R}_t\mathbf{x}_t+\mathbf{t}_t)}$ is removed by considering only the following two constraint equations:

$$\begin{aligned}(\mathbf{C}^{-1}\mathbf{p})^T\hat{\mathbf{x}}^\times\mathbf{C}^{-1}\mathbf{p} = \ & 0 \ = (\mathbf{C}^{-1}\mathbf{p})^T\hat{\mathbf{x}}^\times(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t)\\ (\mathbf{C}^{-1}\mathbf{p})^T\hat{\mathbf{y}}^\times\mathbf{C}^{-1}\mathbf{p} = \ & 0 \ = (\mathbf{C}^{-1}\mathbf{p})^T\hat{\mathbf{y}}^\times(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t) \ .\end{aligned}$$

These equations can be written as (recalling that $\hat{\mathbf{z}}^T\mathbf{C}^{-1}\mathbf{p} = 1$):

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\hat{\mathbf{x}}^T\mathbf{C}^{-1}\mathbf{p} \\ 0 & 1 & -\hat{\mathbf{y}}^T\mathbf{C}^{-1}\mathbf{p} \end{bmatrix}(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t) \ . \tag{2.1}$$

In these equations, the rotation matrix and translation vector are represented as a dual number quaternion. The error metric is simply the square to the equations plus two terms with Lagrange multipliers to enforce the constraints of the dual number quaternions.

The optimization technique is what powers this algorithm. Classic minimization algorithms include steepest descent, Newton's method, and Newton-like methods. The steepest descent minimization algorithms is both inefficient and unreliable because of its slow rate of convergence. Newton's method converges rapidly but is only well defined under certain conditions and, even in under these conditions, convergence is not guaranteed. Two Newton-like methods are in common use in solving other non-linear minimization problems: the Levenberg-Marquardt method and trust-region algorithms [Sor82]. Either could be used to successfully minimize the problem; the paper uses a trust-region algorithm.

For comparison with the other algorithms, the projection constraints of Equation 2.1 will be rewritten by first dividing by $\hat{\mathbf{z}}^T\mathbf{t}_t$ and then rearranging terms:

$$\begin{aligned}\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & -\hat{\mathbf{x}}^T\mathbf{C}^{-1}\mathbf{p} \\ 0 & 1 & -\hat{\mathbf{y}}^T\mathbf{C}^{-1}\mathbf{p} \\ 0 & 0 & 0 \end{bmatrix}(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t)\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t}\\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & -\hat{\mathbf{x}}^T\mathbf{C}^{-1}\mathbf{p} \\ 0 & 1 & -\hat{\mathbf{y}}^T\mathbf{C}^{-1}\mathbf{p} \\ 0 & 0 & 0 \end{bmatrix}\mathbf{R}_t\mathbf{x}_t\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t} + \mathbf{t}_t\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t} - \mathbf{C}^{-1}\mathbf{p}\end{aligned}$$

$$\mathbf{C}^{-1}\mathbf{p} \;=\; \left(\begin{bmatrix} 1 & 0 & -\hat{\mathbf{x}}^T\mathbf{C}^{-1}\mathbf{p} \\ 0 & 1 & -\hat{\mathbf{y}}^T\mathbf{C}^{-1}\mathbf{p} \\ 0 & 0 & 0 \end{bmatrix} \mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t\right)\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t}\;. \tag{2.2}$$

2.1.2 DeMenthon and Davis 1995

This algorithm is based upon using the perspective projection model as an approximation to the weak-perspective projection model. From the perspective (Equation 1.3) and weak-perspective (Equation 1.5) projection equations, the relationship between them is clearly:

$$\mathbf{p}^w = \mathbf{p}\frac{\hat{\mathbf{z}}^T\left(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t\right)}{\hat{\mathbf{z}}^t\mathbf{t}_t} = \mathbf{p}\left(1 + \frac{\hat{\mathbf{z}}^T\mathbf{R}_t\mathbf{x}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t}\right)\;. \tag{2.3}$$

This provides for estimating the weak-perspective projection of the object points given the object pose parameters and the perspective projection.

The algorithm starts by assuming the object is far away:

$$\frac{\hat{\mathbf{z}}^T\mathbf{R}_t\mathbf{x}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t} \approx 0$$

and hence the perspective projection model can be used as an approximation to the weak-perspective projection model directly.

Combining the weak-perspective projection model assuming the translation vector as the reference point (Equation 1.5 assuming $z_o = \hat{\mathbf{z}}^T\mathbf{t}_t$) with the transformation from target-to-camera coordinates assuming the camera's coordinate system is the world coordinate system (Equation 1.2):

$$\mathbf{p}^w = \mathbf{C}\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\left(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t\right)\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right)\;. \tag{2.4}$$

As written, the algorithm assumes the projection of the object frame's origin $(\mathbf{C}\mathbf{t}_t\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t})$ is known. This is not a necessity of the algorithm and, for comparison purposes, the assumption is removed.

The algorithm rewrites Equation 2.4 in terms of values that can be linearly estimated. Define:

$$\begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \end{bmatrix} \equiv \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\mathbf{R}_t\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t}, \quad \alpha \equiv \frac{\hat{\mathbf{x}}^T\mathbf{t}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t}, \quad \text{and} \quad \beta \equiv \frac{\hat{\mathbf{y}}^T\mathbf{t}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t}$$

then Equation 2.4 can be expressed as:

$$\mathbf{p}^w = \mathbf{C}\left(\begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \mathbf{0}^T \end{bmatrix} \mathbf{x}_t + \begin{bmatrix} \alpha \\ \beta \\ 1 \end{bmatrix}\right)$$

To obtain a linear estimation problem, the two constraints on $\mathbf{a}_1$ and $\mathbf{a}_2$ - that the vectors are perpendicular and of identical magnitude - are not maintained.

Given these estimated values, the parameters relevant to obtaining the next iteration's approximation to the weak-perspective projection are computed:

$$\hat{\mathbf{z}}^T\mathbf{t}_t = \frac{1}{2}\left(\frac{1}{|\mathbf{a}_1|} + \frac{1}{|\mathbf{a}_2|}\right) \qquad \mathbf{R}_t^T\hat{\mathbf{z}} = \frac{\mathbf{a}_1 \times \mathbf{a}_2}{|\mathbf{a}_1 \times \mathbf{a}_2|} \ .$$

The algorithm computes values for the weak-perspective projection with Equation 2.3 and then re-estimates the object pose. At convergence, the desired parameters are easily recovered:

$$\hat{\mathbf{z}}^T\mathbf{t}_t = \frac{1}{2}\left(\frac{1}{|\mathbf{a}_1|} + \frac{1}{|\mathbf{a}_2|}\right) \qquad \hat{\mathbf{x}}^T\mathbf{t}_t = \alpha(\hat{\mathbf{z}}^T\mathbf{t}_t) \qquad \hat{\mathbf{y}}^T\mathbf{t}_t = \beta(\hat{\mathbf{z}}^T\mathbf{t}_t)$$

$$\mathbf{R}_t^T\hat{\mathbf{x}} = \frac{\mathbf{a}_1}{|\mathbf{a}_1|} \qquad \mathbf{R}_t^T\hat{\mathbf{y}} = \frac{\mathbf{a}_2}{|\mathbf{a}_2|} \qquad \mathbf{R}_t^T\hat{\mathbf{z}} = \frac{\mathbf{a}_1 \times \mathbf{a}_2}{|\mathbf{a}_1 \times \mathbf{a}_2|} \ .$$

Note that the recovered rotation matrix need not be orthogonal. The paper does not describe a method for enforcing the orthogonality while performing the minimization.

The above algorithm assumes a three-dimensional target. In a paper by Oberkampf, DeMenthon, and Davis [ODD93], the algorithm is extended to planar objects. For a planar object, only the first two components of the vectors $\mathbf{R}_t^T\hat{\mathbf{x}}\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t}$ and $\mathbf{R}_t^T\hat{\mathbf{y}}\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t}$ are estimated. Given these four values, the other two must be computed. This is done by enforcing the two constraints of the scaled rotation vectors; they are perpendicular and of identical magnitude. This creates a non-linear system of equations which generally has two solutions; as expected. Looking at the weak-perspective projection model, it is clear that a plane and its 'reflection' through the reference plane have the same set of projected points. The iterative process is modified as follows. Compute the initial two possible solutions. For each solution, perform an iteration to improve the estimate and keep the estimate with lower residual error as measured in the image plane. Maintain two solutions until each reaches convergences and then choose the one with lesser residual in the image plane. Figure 2.3 depicts this method.

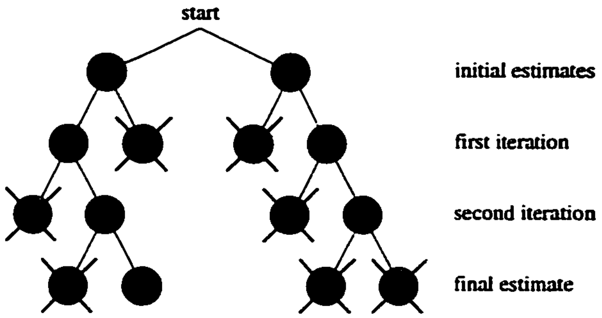


Figure 2.3: Planar Object Pose Computation Tree.

2.1.3 Horaud, Dornaika, Lamiroy, and Christy 1997

This algorithm is very similar to the previous except that the perspective projection model is used as an approximation to the paraperspective projection model. From the perspective (Equation 1.3) and paraperspective (Equation 1.4) projection equations, the relationship between them is:

$$\mathbf{p}^p = \mathbf{p}\frac{\hat{\mathbf{z}}^T(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t)}{\hat{\mathbf{z}}^T\mathbf{t}_t} - \mathbf{t}_t\hat{\mathbf{z}}^T\mathbf{R}_t\mathbf{x}_t\frac{1}{(\hat{\mathbf{z}}^T\mathbf{t}_t)^2} \; . \tag{2.5}$$

This provides for estimating the paraperspective projection of the object points given the object pose parameters and the perspective projection.

Similar to the previous algorithm, the object is initially assumed far away:

$$\frac{\hat{\mathbf{z}}^T\mathbf{R}_t\mathbf{x}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t} \approx 0$$

and hence the perspective projection model can be used as an approximation to the paraperspective projection model.

Combining the paraperspective projection model assuming the translation vector as the reference point (Equation 1.4 assuming $\mathbf{x}_o = \mathbf{t}_t$) with the transformation from target-to-camera coordinates assuming the camera's coordinate system is the world coordinate system (Equation 1.2):

$$\mathbf{p}^p = \mathbf{C}\left(\begin{bmatrix} 1 & 0 & -\frac{\hat{\mathbf{x}}^T\mathbf{t}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t} \\ 0 & 1 & -\frac{\hat{\mathbf{y}}^T\mathbf{t}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t} \\ 0 & 0 & 0 \end{bmatrix}(\mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t)\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t} + \begin{bmatrix} \frac{\hat{\mathbf{x}}^T\mathbf{t}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t} \\ \frac{\hat{\mathbf{y}}^T\mathbf{t}_t}{\hat{\mathbf{z}}^T\mathbf{t}_t} \\ 1 \end{bmatrix}\right) \; . \tag{2.6}$$

As written, the algorithm assumes the projection of the object frame's origin ($\mathbf{C}\mathbf{t}_t\frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t}$) is known. This is not a necessity of the algorithm and, for comparison purposes, the assumption is removed.

The algorithm rewrites Equation 2.6 in terms of values that can be linearly estimated. Define:

$$\begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \end{bmatrix} \equiv \begin{bmatrix} 1 & 0 & -\frac{\hat{\mathbf{x}}^T \mathbf{t}_t}{\hat{\mathbf{z}}^T \mathbf{t}_t} \\ 0 & 1 & -\frac{\hat{\mathbf{y}}^T \mathbf{t}_t}{\hat{\mathbf{z}}^T \mathbf{t}_t} \end{bmatrix} \mathbf{R}_t \frac{1}{\hat{\mathbf{z}}^T \mathbf{t}_t}, \quad \alpha \equiv \frac{\hat{\mathbf{x}}^T \mathbf{t}_t}{\hat{\mathbf{z}}^T \mathbf{t}_t}, \quad \text{and} \quad \beta \equiv \frac{\hat{\mathbf{y}}^T \mathbf{t}_t}{\hat{\mathbf{z}}^T \mathbf{t}_t}$$

then Equation 2.6 can be expressed as:

$$\mathbf{p}^p = \mathbf{C} \left(\begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \mathbf{0}^T \end{bmatrix} \mathbf{x}_t + \begin{bmatrix} \alpha \\ \beta \\ 1 \end{bmatrix} \right)$$

To obtain a linear estimation problem, the constraints on $\mathbf{a}_1$ and $\mathbf{a}_2$ are not maintained.

Given these estimated values, the parameters relevant to obtaining the next iteration's approximation to the paraperspective projection are computed:

$$\hat{\mathbf{z}}^T \mathbf{t}_t = \frac{1}{2} \left(\frac{\sqrt{1+\alpha^2}}{|\mathbf{a}_1|} + \frac{\sqrt{1+\beta^2}}{|\mathbf{a}_2|} \right) .$$

Estimating $\mathbf{R}_t^T \hat{\mathbf{z}}$ is not trivial. A system of equations is derived and their solution provides the desired vector. Notationally, $\mathbf{a}_1^\times$ represents the skew-symmetric matrix equivalent to the cross-product operator.

$$\begin{aligned} \mathbf{R}_t^T \hat{\mathbf{z}} &= (\mathbf{R}_t^T \hat{\mathbf{x}}) \times (\mathbf{R}_t^T \hat{\mathbf{y}}) \\ &= \left(\mathbf{a}_1 \hat{\mathbf{z}}^T \mathbf{t}_t + \mathbf{R}_t^T \hat{\mathbf{z}} \alpha \right) \times \left(\mathbf{a}_2 \hat{\mathbf{z}}^T \mathbf{t}_t + \mathbf{R}_t^T \hat{\mathbf{z}} \beta \right) \\ \left(\mathbf{I} - \beta(\hat{\mathbf{z}}^T \mathbf{t}_t) \mathbf{a}_1^\times + \alpha(\hat{\mathbf{z}}^T \mathbf{t}_t) \mathbf{a}_2^\times \right) \mathbf{R}_t^T \hat{\mathbf{z}} &= \mathbf{a}_1 \times \mathbf{a}_2 (\hat{\mathbf{z}}^T \mathbf{t}_t)^2 \end{aligned}$$

This system of equations is always solvable [HDLC97].

The algorithm computes values for the paraperspective projection with Equation 2.5 and then re-estimates the object pose. At convergence, the desired parameters are recovered using the above for $\hat{\mathbf{z}}^T \mathbf{t}_t$ and $\mathbf{R}_t^T \hat{\mathbf{z}}$ plus:

$$\begin{aligned} \hat{\mathbf{x}}^T \mathbf{t}_t &= \alpha(\hat{\mathbf{z}}^T \mathbf{t}_t) & \hat{\mathbf{y}}^T \mathbf{t}_t &= \beta(\hat{\mathbf{z}}^T \mathbf{t}_t) \\ \mathbf{R}_t^T \hat{\mathbf{x}} &= \frac{\mathbf{a}_1(\hat{\mathbf{z}}^T \mathbf{t}_t) + (\mathbf{R}_t^T \hat{\mathbf{z}})\alpha}{|\mathbf{a}_1(\hat{\mathbf{z}}^T \mathbf{t}_t) + (\mathbf{R}_t^T \hat{\mathbf{z}})\alpha|} & \mathbf{R}_t^T \hat{\mathbf{y}} &= \frac{\mathbf{a}_2(\hat{\mathbf{z}}^T \mathbf{t}_t) + (\mathbf{R}_t^T \hat{\mathbf{z}})\beta}{|\mathbf{a}_2(\hat{\mathbf{z}}^T \mathbf{t}_t) + (\mathbf{R}_t^T \hat{\mathbf{z}})\beta|} \end{aligned}$$

Again note that the recovered rotation matrix need not be orthogonal. To orthogonalize the matrix, the paper suggests estimating the rotation matrix closest in value to the three vectors $\mathbf{R}_t^T\hat{\mathbf{x}}$, $\mathbf{R}_t^T\hat{\mathbf{y}}$, and $\mathbf{R}_t^T\hat{\mathbf{z}}$ with the metric:

$$\min_{\mathbf{R}} \left(\left|\mathbf{R}(\mathbf{R}_t^T\hat{\mathbf{x}}) - \hat{\mathbf{x}}\right|^2 + \left|\mathbf{R}(\mathbf{R}_t^T\hat{\mathbf{y}}) - \hat{\mathbf{y}}\right|^2 + \left|\mathbf{R}(\mathbf{R}_t^T\hat{\mathbf{z}}) - \hat{\mathbf{z}}\right|^2 \right)$$

This minimization has a closed-form solution [Fau93]. However, this minimization need not provide the **R** that minimizes the pose estimation problem nor even provide a pose that - if used during the iterations - improves the metric fit. As such, this is a poor method.

As described above, the algorithm is for three-dimensional objects. A solution similar to the planar problem solution in Section 2.1.2 is presented in the paper using the same technique of maintaining two possible solutions.

2.2 Proposed Pose Estimation Method

2.2.1 Introduction

Equations 1.4, 1.5, and 2.2 provide three simplifications of the perspective projection model. On the surface, the only difference between the three sets of equations are the measurements on the left hand side and how the point is projected onto the frontal-parallel plane through the translation vector. If there were no localization error, Equation 2.2 correctly projects the points to the reference plane and then to the sensor. In the presence of measurement noise, Equation 2.2 can be interpreted two ways. The first is that the noisy measurements are coupled with the known data; this would lead to a very difficult estimation problem which is clearly not solved in the literature. The second is that the projection is an approximation to perspective projection where, instead of one vector being used to project all the points to the frontal-parallel plane, a set of vectors are used to project the points to the frontal-parallel plane. If these points are closer to the 'true' projection than the projection of the translation vector, a better approximation to the perspective projection equations is obtained relative to paraperspective projection. Similar to the weak-perspective and paraperspective projection approximation algorithms, this projection approximation can be iteratively improved by updating the estimates of the projected points.

Also, there is a greater need to constrain the rotation matrix correctly in Equation 2.2

than in Equation 1.4 and Equation 1.5. If the matrix is unconstrained, Equation 2.2 uses nine unknowns to represent four values while Equation 1.4 and Equation 1.5 use only six unknowns for four values.

The projection approximation algorithms have two weaknesses that can be readily identified and improved: distortion of the true perspective projection error metric and maintenance of the rotation matrix constraints. Techniques for improving these deficits as well as using the better projection approximation Equation 2.2 are presented.

2.2.2 Simulation Method

Before evaluating and comparing the algorithms in a quantitative manner, the framework for comparison must be defined.

The object used in the simulations is a cube of width one hundred millimeters. The eight corners of the cube plus the centroid of the cube are the data points and the object is assumed to be a wire-frame so that all nine points are always imaged. The correspondence between the model points and the image points is assumed known.

The camera used to image the object had a focal length of 8.5 millimeters. The imaging device has dimensions of 320 × 240 pixels; one pixel is equivalent to 0.0275 millimeters.

The object is given a random rotation by choosing an axis of rotation and an angle. The axis is chosen by first taking a vector comprised of three components selected from a uniform distribution over the range $[0, 1)$ and then the vector is normalized. The angle is chosen from a uniform distribution over $[-\pi, \pi)$.

The depth of the object's transformed origin is the parameter varied over in the simulations. The other two components of the translation vector are chosen from a uniform distribution over the imaging device, and scaled by the depth divided by the focal length.

The projection of the points is then performed. If any point of the object is not on the imaging sensor, the translation and rotation parameters are reselected.

The localization error is assumed to be Gaussian with a standard deviation set in the simulation (0.2, 0.5, or 1.0 pixels). The noise is added in the plane containing the imaging device.

Finally, each point is divided by the focal length to provide the measurements used in pose estimation.

In the literature, two metrics for determining goodness-of-fit are used; relative position error and orientation error. Relative position error is defined as taking the magnitude of the vector between the true and estimated translation vectors and dividing by the magnitude of the true translation vector. Orientation error is computed by finding the angle of the rotation between the true and estimated rotation matrices. For this purpose, the resultant rotation matrices of the algorithms are always orthogonalized.

For each depth value, fifty thousand trials are performed and the average for both metrics is reported. For each algorithm at each depth, twenty iterations are performed in the minimization; this is significantly more than is needed for convergence.

Along with absolute results, graphs representing comparisons between algorithms are presented; these graphs have line labels such as "polished / weak". This denotes that the graph is of the relevant metric value produced by the polished algorithm divided by the metric value for the weak-perspective projection algorithm.

2.2.3 Reducing Error Metric Distortion

In both the weak-perspective and paraperspective algorithms, the projection approximation is calculated by multiplying the noisy image measurement by a factor and then, in the paraperspective case, adding an offset. Clearly, the noise in each point of the approximation projection model has been scaled and, by not removing this effect, the estimation is inappropriately biased towards reducing the error at points further away. For both algorithms, this can be accomplished by simply multiplying every equation by:

$$\frac{\hat{\mathbf{z}}^T \mathbf{t}_t}{\hat{\mathbf{z}}^T \mathbf{R}_t \mathbf{x}_t + \hat{\mathbf{z}}^T \mathbf{t}_t}$$

before squaring for the error metric. Since the change to the metric is more significant when points are closer, it is expected that the improvement in performance will be more significant when the object distance to size ratio is small.

While this does not completely remove the error metric distortion, during simulations it improved the orientation and position metric results for both algorithms in the shortest distances by about ten percent.

2.2.4 Polishing the Estimate

If the error-free values of the imaged data were known, the error metric represented by Equation 2.2 can be modified so as to be ideal in terms of providing a metric for pose estimation. Possibly, by using sufficiently good estimates - by having good initial values for the pose estimation - of these imaged values, the modified metric can be used to iteratively obtain the true pose estimation parameters. Without a good initial guess, like most other non-linear algorithms, this algorithm will fail. From the results seen for the modified paraperspective algorithm, it is clear that the pose estimated by that algorithm can be used as an initial estimate.

Furthermore, if the residual rotation between the current estimate and the true value is small enough, a linearization of the rotation matrix in the estimation problem can be used. From the simulation results, this is clearly the case. The new estimate of the rotation matrix, $\mathbf{R}_t$, will be approximated by applying an approximation of a small rotation matrix to the current estimate of the rotation matrix $\mathbf{R}_{t_0}$:

$$\mathbf{R}_t \approx \begin{bmatrix} 1 & -w_z & w_y \\ w_z & 1 & -w_x \\ -w_y & w_x & 1 \end{bmatrix} \mathbf{R}_{t_0}$$

The estimated rotation matrix is orthogonalized after each estimation.

Labeling the current estimate of the projection of the object points as $\mathbf{p}_0$, the error metric is derived from:

$$\mathbf{C}^{-1}\mathbf{p} = \left(\begin{bmatrix} 1 & 0 & -\hat{\mathbf{x}}^T\mathbf{C}^{-1}\mathbf{p}_0 \\ 0 & 1 & -\hat{\mathbf{y}}^T\mathbf{C}^{-1}\mathbf{p}_0 \end{bmatrix} \mathbf{R}_t\mathbf{x}_t + \begin{bmatrix} \hat{\mathbf{x}}^T\mathbf{t}_t \\ \hat{\mathbf{y}}^T\mathbf{t}_t \end{bmatrix} \right) \frac{1}{\hat{\mathbf{z}}^T\mathbf{t}_t} \tag{2.7}$$

In the graphs, the modified paraperspective algorithm was executed for ten iterations and then this 'polished' algorithm was run for ten iterations.

By using the polishing technique, the orientation error for the modified paraperspective algorithm is decreased by over ten percent at all depths. The results for position showed essentially no change.

2.3 Conclusion

The projection approximation algorithms reviewed in this section are viable techniques for object pose estimation. They are reasonably accurate and, relative to non-linear techniques, very fast. A simple modification to the algorithms was presented that removes a large portion of the error observed when the object is close. Finally, a polishing technique was suggested that dramatically improves the accuracy of the estimate. The overall improvements to the algorithms are shown in Figures 2.4, 2.5, and 2.6. The polished algorithm does not require a significant increase in execution time and, as such, is believed to be a complete improvement over the reviewed algorithms. Furthermore, the observed accuracy results suggest that the algorithm can be used in nearly all applications.

However, the initial arguments against using camera calibration and then successive pose estimations to solve the overall problem of estimating camera intrinsic parameters and trajectory still hold: the impracticality of using a single target in a large workspace and the inaccuracy of camera calibration from a small data set.

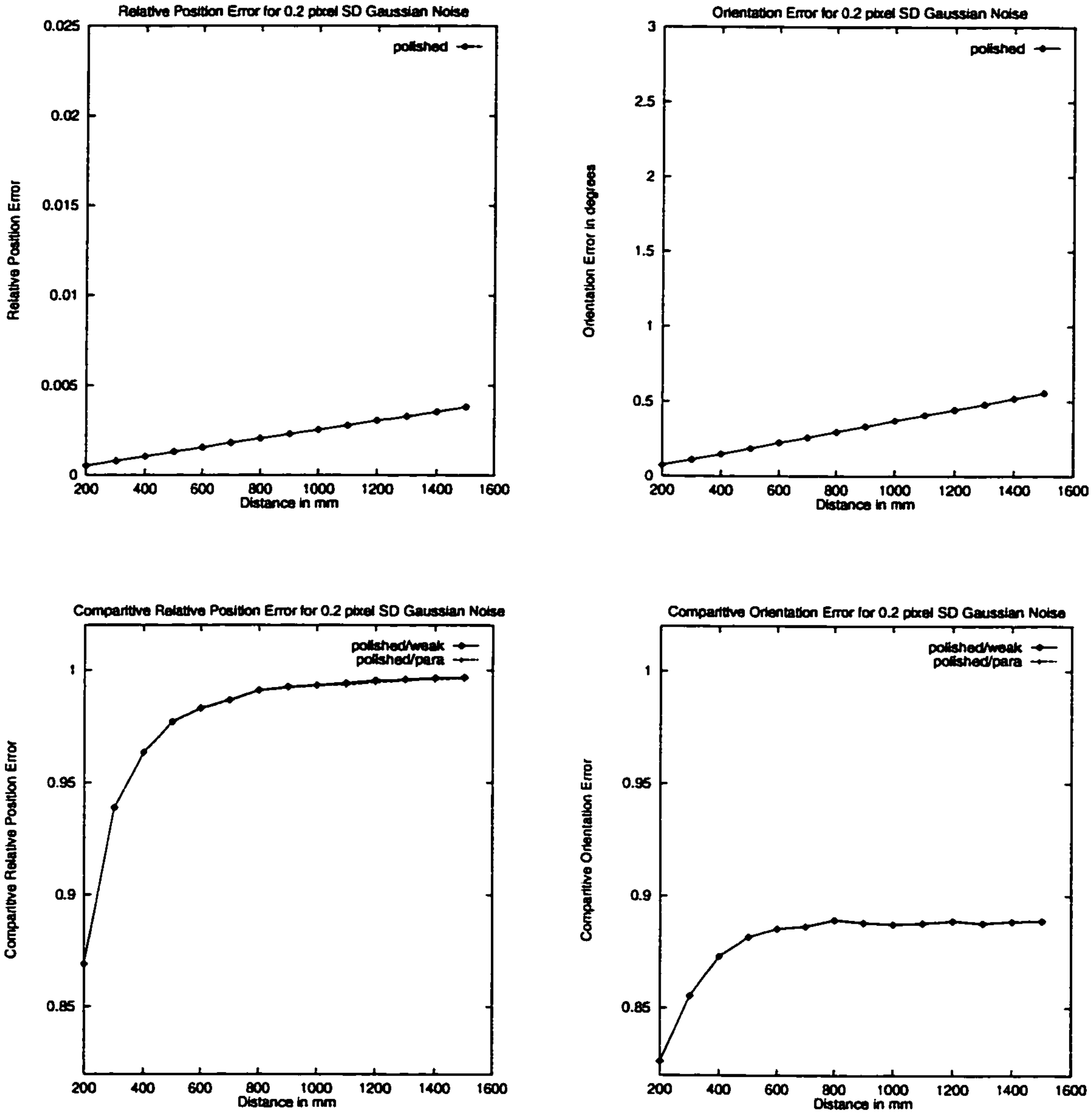


Figure 2.4: Absolute and comparative results for 0.2 pixels of standard deviation. The two curves in both comparative graphs are very close.

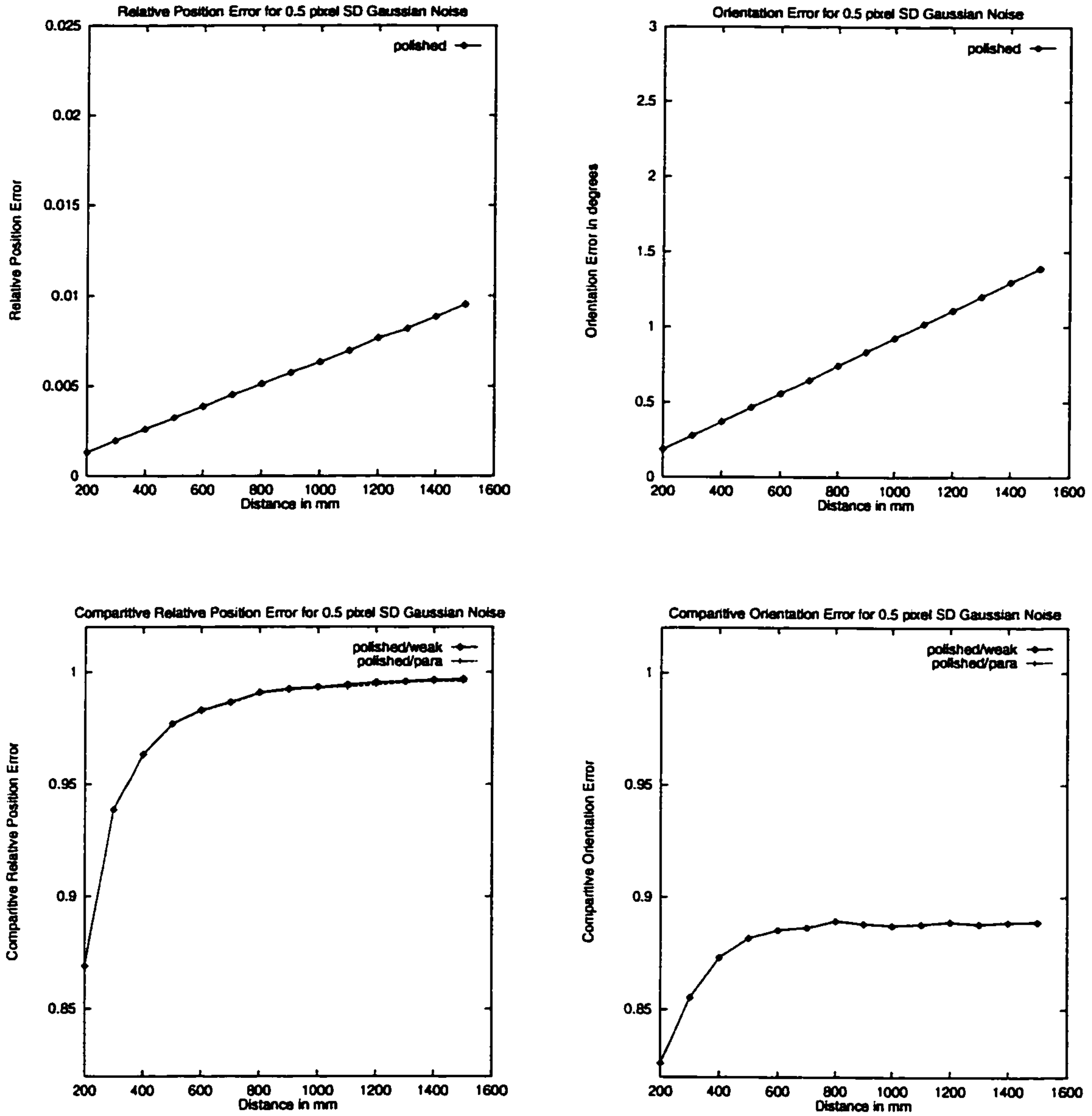


Figure 2.5: Absolute and comparative results for 0.5 pixels of standard deviation. The two curves in both comparative graphs are very close.

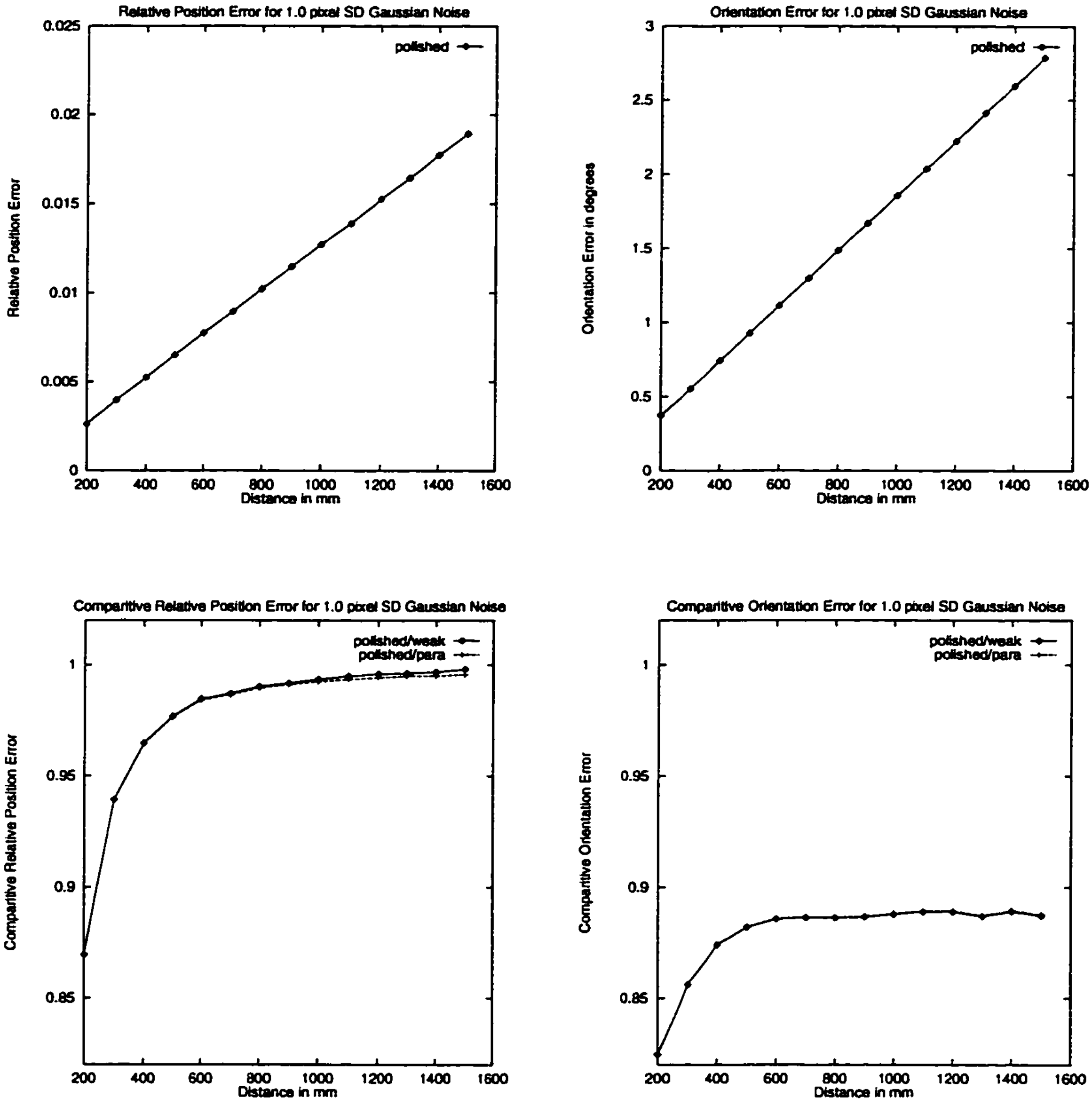


Figure 2.6: Absolute and comparative results for 1.0 pixels of standard deviation. The two curves in both comparative graph are very close.

Chapter 3

Self-Calibration

3.1 Overview

Self-calibration has the same goal of standard calibration but does not rely on the use of a known target. Algorithms use a set of correspondences - typical points or lines - over a set of images to estimate the camera's intrinsic parameters (e.g. focal length and image center). Typically, the location of these fiducials and the motion of the observer are estimated as well (Figure 3.1).

A self-calibration algorithm is given by S. Maybank and O. Faugeras in [MF92]. They investigate the constraints placed upon the absolute conic by epipolar constraints. The absolute conic is a particular conic in the plane at infinity which is invariant under the rigid motions of space. Hence, the image of the absolute conic is determined exclusively by the intrinsic calibration parameters. Rigid motion of the camera constrains the projection into the second image of the inverse projection of a point in the first image to lie along a particular line in the second image. This is the epipolar constraint. Each estimated epipolar transformation imposes two constraints on the projection of the absolute conic. Since a general conic in a plane has five degrees of freedom, the set of calibration parameters consistent with two epipolar transformations is an algebraic curve. Three transformations over-constrain the calibration parameters. Thereby, the intrinsic parameters of a moving camera can be estimated by computing the epipolar geometry over four frames. In the conclusion of the paper, the following two sentences captures the essence of self-calibration:

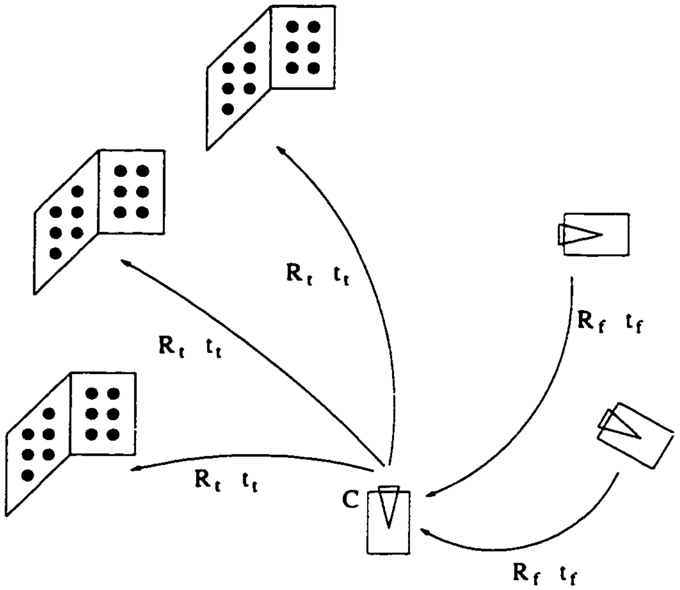


Figure 3.1: Depiction of the unknowns in self-calibration. They are the camera intrinsics $\mathbf{C}$, the motions of the observer described by rotation $\mathbf{R}_f$ and translation $\mathbf{t}_t$, and the rotation $\mathbf{R}_t$ and translation $\mathbf{t}_t$ of the targets.

> It seems a little strange at first that it should be possible to calibrate the camera just by pointing it at the environment, selecting a few points of interest, and tracking them in the image while moving the camera with a motion that does not need to [be] known. But when one thinks of the problem a little more it becomes apparent that the correspondences that are established by the tracking define the epipolar transformations which in turn constrain the intrinsic parameters of the camera, that is, the image of the absolute conic which, by definition so to speak, is invariant under the camera motion (that is why the knowledge of the motion is not necessary).

In search of higher accuracy, Hartley approaches the self-calibration problem by using a large number of images in a non-linear estimation [Har93].

A common sub-field of self-calibration is weak-calibration. Weak-calibration is the estimation of epipolar geometry between two or more images without regard to decomposing this relationship into intrinsic and extrinsic parameters. The Fundamental Matrix [LF96] and Trifocal Tensor [Sha95] are the two most well known foci of weak-calibration; the Fundamental Matrix defines the relationship between two views and the Trifocal Tensor provides constraints over three views (for analysis of common metrics for estimating the Fundamental Matrix see [Zha98]). Constraints over more than three images are typically not investigated since these can be generated by linear combinations of the Fundamental Matrices and Trifocal Tensors constraints over the images [FM95].

Optical flow - the estimation of the motion of each pixel from one image to another - can be used to provide the correspondences needed for estimating the weak-calibration parameters. Furthermore, using the framework of Bergen et. al. in [BAHH92], the simultaneous estimation of weak-calibration and optical flow is possible. Presented here is an algorithm for estimating the Fundamental Matrix and the optical flow constrained to that matrix.

3.2 Discrete-Time Rigidity-Constrained Optical Flow

3.2.1 Introduction

Estimation of optical flow, a longstanding problem in computer vision, is particularly difficult when the displacements are large. Multi-scale algorithms can handle large image displacements and also improve overall accuracy of optical flow fields [Ana89, EN88, LKW92, SAH91]. However, these techniques typically make the unrealistic assumption that the flow field is smooth. In many situations, a more plausible assumption is that of a rigid world.

Given point and/or line correspondences, the discrete-time rigid motion problem has been studied and solved by a number of authors (e.g. [HGC92, Har94, LV94, SN94, VZR95]). For instantaneous representations, multi-scale estimation techniques have been used to couple the flow and motion estimation problems to provide a direct method for planar surfaces [BAHH92, Hor86]. These methods use the multi-scale technique to capture large motions while significantly constraining the flow with a global model. But the planar world assumption is quite restrictive, and the approach also contains a hidden contradiction; the algorithm can observe large image motions but can only represent small camera motions due to the instantaneous time assumption.

This section describes an optical flow algorithm for discrete camera motion in a rigid world. The algorithm is based on differential image measurements and estimates are computed within a multi-scale decomposition; the estimates are propagated from coarse to fine scales. Unlike traditional coarse-to-fine approaches which impose smoothness on the flow field, this algorithm assumes smoothness of the inverse depth values.

3.2.2 Discrete-Time Optical Flow

The imaging system is assumed to use the following projection model:

$$\mathbf{p}_i = \mathbf{C}\mathbf{x}_i \frac{1}{z_i} \quad \text{where} \quad \mathbf{x}_i \equiv \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \quad \text{and} \quad \mathbf{p}_i \equiv \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \; ; \tag{3.1}$$

$\mathbf{x}_i$ denotes the point's coordinates in the camera's frame of reference and $\mathbf{p}_i$ the image coordinates. The matrix $\mathbf{C}$ contains the camera calibration parameters and is presumed invertible. This assumption is valid for any reasonable camera system. For example, the pin-hole model is included in this family, as well as more complex models such as that given in [VF96].

The discrete motion of a point is expressed as:

$$\mathbf{x'}_i = \mathbf{R}\mathbf{x}_i + \mathbf{t} \ , \tag{3.2}$$

where $\mathbf{R}$ is a (discrete-time) rotation matrix, $\mathbf{t}$ is a translation vector, and $\mathbf{x'}_i$ denotes the point's coordinates after the discrete motion.

A classic formulation of this constraint is due to Longuet-Higgins [LHP80]:

$$\mathbf{x}'^T_i \left(\mathbf{t} \times \mathbf{R}\mathbf{x}_i\right) = 0 \ .$$

Using equation (3.1) to substitute for $\mathbf{x}_i$ gives:

$$z'_i \mathbf{p}'^T_i (\mathbf{C}'^{-1})^T \left(\mathbf{t} \times \mathbf{R}\mathbf{C}^{-1}\mathbf{p}_i z_i\right) = 0 \ . \tag{3.3}$$

Let $\mathbf{t}^\times$ represent the skew-symmetric matrix corresponding to a cross-product with $\mathbf{t}$. Using suitable linear algebraic identities, and assuming that $z_i \neq 0$ and $z'_i \neq 0$, leads to the following simplification:

$$0 \;=\; z'_i \mathbf{p}'^T_i (\mathbf{C}'^{-1})^T \mathbf{t}^\times \mathbf{R}\mathbf{C}^{-1}\mathbf{p}_i z_i$$

$$\begin{aligned} 0 &= \mathbf{p'}_i^T(\mathbf{C'}^{-1})^T\mathbf{t}^\times\mathbf{C'}^{-1}\mathbf{C'}\mathbf{R}\mathbf{C}^{-1}\mathbf{p}_i \\ 0 &= \mathbf{p'}_i^T(\mathbf{C'}\mathbf{t})^\times\mathbf{C'}\mathbf{R}\mathbf{C}^{-1}\mathbf{p}_i \\ 0 &= \mathbf{p'}_i^T\mathbf{F}\mathbf{p}_i \ , \end{aligned} \tag{3.4}$$

where $\mathbf{F} \equiv (\mathbf{C'}\mathbf{t})^\times\mathbf{C'}\mathbf{R}\mathbf{C}^{-1}$ is a matrix that depends on the global motion and camera calibration information. Equation (3.4) provides a constraint on the initial and final image positions assuming rigid-body motion and is the fundamental matrix [LF96].

In addition, it will be useful to develop an expression for the final position, $\mathbf{p'}_i$, given the calibration, motion, and structure parameters. Substituting the inverse of equation (3.1) into the rigid-body motion constraint of equation (3.2):

$$\mathbf{C'}^{-1}\mathbf{p'}_i z'_i = \mathbf{R}\mathbf{C}^{-1}\mathbf{p}_i z_i + \mathbf{t} \ .$$

Solving for the image position after the motion:

$$\mathbf{p'}_i = \frac{\mathbf{C'}\mathbf{R}\mathbf{C}^{-1}\mathbf{p}_i + \mathbf{C'}\mathbf{t}\frac{1}{z_i}}{\hat{\mathbf{z}}^T\left(\mathbf{C'}\mathbf{R}\mathbf{C}^{-1}\mathbf{p}_i + \mathbf{C'}\mathbf{t}\frac{1}{z_i}\right)} \quad \text{where} \quad \hat{\mathbf{z}} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \ . \tag{3.5}$$

This rigid-world motion constraint must be connected with measurements of image displacements. Since differential optical flow techniques have proven to be quite robust [BFB92], the formulation is based on the differential form of the 'brightness constancy constraint' [Hor86]:

$$\left.\frac{\partial I}{\partial u}\right|_i \cdot \frac{\partial u_i}{\partial t} + \left.\frac{\partial I}{\partial v}\right|_i \cdot \frac{\partial v_i}{\partial t} + \left.\frac{\partial I}{\partial t}\right|_i = 0 \ .$$

Substituting discrete displacements for the differential changes in image positions - the displacements are assumed to be small given the multi-scale (coarse-to-fine) framework described in Section 3.2.3 - and rewriting to isolate $\mathbf{p}'_i$ gives:

$$\begin{bmatrix} \partial I/\partial u \\ \partial I/\partial v \\ \partial I/\partial t \end{bmatrix}_i^T \begin{bmatrix} u'_i - u_i \\ v'_i - v_i \\ 1 \end{bmatrix} = \begin{bmatrix} \partial I/\partial u \\ \partial I/\partial v \\ \partial I/\partial t \end{bmatrix}_i^T \begin{bmatrix} 1 & 0 & -u_i \\ 0 & 1 & -v_i \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}'_i = 0 \ . \tag{3.6}$$

This constraint is combined with equation (3.5), squared, and summed over a local neighborhood, $N(i)$, to produce an error metric:

$$E_i(\mathbf{A}, \mathbf{b}, \frac{1}{z_i}) = \frac{\left(\mathbf{A}\mathbf{p}_i + \mathbf{b}\frac{1}{z_i}\right)^T \mathbf{D}_i \left(\mathbf{A}\mathbf{p}_i + \mathbf{b}\frac{1}{z_i}\right)}{\left(\hat{\mathbf{z}}^T\mathbf{A}\mathbf{p}_i + \hat{\mathbf{z}}^T\mathbf{b}\frac{1}{z_i}\right)^2} \ ,$$

where $\mathbf{A} \equiv \mathbf{C}'\mathbf{R}\mathbf{C}^{-1}$, $\mathbf{b} \equiv \mathbf{C}'\mathbf{t}$, and $\mathbf{D}_i$ is a matrix constructed from the differential image measurements and known position vectors:

$$\mathbf{D}_i \equiv \begin{bmatrix} 1 & 0 & -u_i \\ 0 & 1 & -v_i \\ 0 & 0 & 1 \end{bmatrix}^T \left(\sum_{j \in \mathcal{N}(i)} \begin{bmatrix} \partial I/\partial u \\ \partial I/\partial v \\ \partial I/\partial t \end{bmatrix}_j \begin{bmatrix} \partial I/\partial u \\ \partial I/\partial v \\ \partial I/\partial t \end{bmatrix}_j^T \right) \begin{bmatrix} 1 & 0 & -u_i \\ 0 & 1 & -v_i \\ 0 & 0 & 1 \end{bmatrix} ,$$

Minimizing $E_i(\mathbf{A}, \mathbf{b}, \frac{1}{z_i})$ with respect to $\frac{1}{z_i}$ gives:

$$\frac{1}{z_i} = -\frac{\mathbf{p}_i^T \mathbf{A}^T \mathbf{D}_i \left(\mathbf{I} - \mathbf{b}\hat{\mathbf{z}}^T\right) \mathbf{A}\mathbf{p}_i}{\mathbf{b}^T \mathbf{D}_i \left(\mathbf{I} - \mathbf{b}\hat{\mathbf{z}}^T\right) \mathbf{A}\mathbf{p}_i} .$$

Substituting back into $E_i(\mathbf{A}, \mathbf{b}, \frac{1}{z_i})$, and noting that $\mathbf{F} = \mathbf{b}^\times \mathbf{A}$:

$$E_i(\mathbf{A}, \mathbf{b}) = -\frac{\mathbf{b}^T \mathbf{D}_i \left(\mathbf{F}\mathbf{p}_i\right)^\times \mathbf{D}_i \mathbf{A}\mathbf{p}_i}{\left(\mathbf{F}\mathbf{p}_i\right)^T (\hat{\mathbf{z}}^\times)^T \mathbf{D}_i \hat{\mathbf{z}}^\times \left(\mathbf{F}\mathbf{p}_i\right)} .$$

Using a series of linear algebraic manipulations, the numerator may be rewritten as an expression quadratic in $\mathbf{F}$:

$$\begin{aligned} -(\mathbf{D}_i\mathbf{b})^T(\mathbf{F}\mathbf{p}_i)^\times \mathbf{D}_i\mathbf{A}\mathbf{p}_i &= (\mathbf{F}\mathbf{p}_i)^T(\mathbf{D}_i\mathbf{b})^\times \mathbf{D}_i\mathbf{A}\mathbf{p}_i \\ &= (\mathbf{F}\mathbf{p}_i)^T \mathit{Adj}(\mathbf{D}_i)\mathbf{b}^\times \mathbf{A}\mathbf{p}_i \\ &= (\mathbf{F}\mathbf{p}_i)^T \mathit{Adj}(\mathbf{D}_i)\mathbf{F}\mathbf{p}_i \ , \end{aligned}$$

where $\mathit{Adj}(\mathbf{D}_i)$ indicates the adjoint of the matrix $\mathbf{D}_i$.

To efficiently solve for $\mathbf{F}$, a global metric is formed by summing over the image the weighted numerators of the $E_i(\mathbf{A}, \mathbf{b})$:

$$E(\mathbf{A}, \mathbf{b}) = \sum_i (\mathbf{F}\mathbf{p}_i)^T \mathit{Adj}(\mathbf{D}_i)\mathbf{F}\mathbf{p}_i / w_i \ , \tag{3.7}$$

where w_i is the value of the denominator of $E_i(\mathbf{A}, \mathbf{b})$ using the previous estimate of $\mathbf{F}$. This metric is computed iteratively in the coarse-to-fine procedure and, from empirical observations, only one iteration at each scale is necessary.

The algorithm proceeds by first globally minimizing equation (3.7) to obtain a solution for the nine entries of the matrix $\mathbf{F}$, subject to the constraints $|\mathbf{F}| = 0$ and $\sum_j \sum_i (\mathbf{L}_{j,i})^2 = 1$ (to remove the scale ambiguity). Then, the squared optical flow constraint

$$E_f(\mathbf{p}'_i) = \mathbf{p}'^T_i \mathbf{D}_i \mathbf{p}'_i \ , \tag{3.8}$$

is minimized at each pixel, subject to equation (3.4) with the estimated value of $\mathbf{F}$, to obtain an optical flow field.

3.2.3 Multi-Scale Implementation

Since the method is capable of estimating large (discrete) camera motions, it must be able to handle large image displacements. This is accomplished with a coarse-to-fine version of the algorithm on a multi-scale decomposition.

First, a Gaussian pyramid is constructed on the pair of input frames [Bur81]. At the coarsest scale of the pyramid, the algorithm is employed as derived in the previous section to provide an initial coarse estimate of optical flow. This optical flow is interpolated to give a finer resolution flow field, denoted $(\Delta^c u_i, \Delta^c v_i)$. This motion is removed from the finer scale images using warping; the warped images are denoted I^w.

Since the optical flow equation (3.8) is written only in terms of the final positions $\mathbf{p}'_i$, the constraint on the warped images needs only a slight modification:

$$\begin{bmatrix} \partial I^w/\partial u \\ \partial I^w/\partial v \\ \partial I^w/\partial t \end{bmatrix}_i^T \begin{bmatrix} 1 & 0 & -(u_i + \Delta^c u_i) \\ 0 & 1 & -(v_i + \Delta^c v_i) \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}'_i = 0 \ .$$

The remainder of the algorithm is as before: new matrices $\mathbf{D}_i$ are computed from this constraint and these are used to estimate $\mathbf{F}$ using equation (3.7). The weightings, w_i, are computed using the estimate of $\mathbf{F}$ from the previous scale. Finally, equation (3.8) is minimized at each pixel, subject to the constraint of equation (3.4), to estimate the optical flow.

3.2.4 Experimental Results

Experimental results were collected for three different algorithms on three sequences. The first method is a simple multi-scale optical flow (msof) algorithm [SAH91]. The second computes flow for discrete motion of a planar world (planar) [MSB97]. The third is the algorithm presented in this paper (rigid).

The first sequence is the 'Yosemite' sequence which was graphically rendered from an aerial photograph and range map by Lyn Quam at SRI. True optical flow vectors were computed from the motion, structure, and calibration data provided with the sequence. The textureless top region was ignored during error calculations. In order to obtain large

sequence →	**Yosemite**						
interval →	**1**	**2**	**3**	**4**	**5**	**6**	**7**
msof	6.13^o	7.49^o	13.10^o	21.32^o	29.09^o	35.82^o	42.43^o
planar	5.82^o	6.56^o	6.70^o	6.70^o	6.92^o	6.82^o	7.63^o
rigid	5.77^o	5.86^o	6.18^o	6.53^o	6.55^o	6.77^o	6.95^o

Table 3.1: Mean angular error in flow vectors for three different algorithms for the Yosemite sequence. The interval refers to the temporal sampling of the sequence (e.g. every frame was used, every second frame, every third, etc.).

motions, the computations are performed on the sequence subsampled at different temporal rates.

The second and third sequence were taken in the GRASP Laboratory from a camera mounted on a tripod. Six markers consisting of seven black disks on a white planar surface were placed in the scene and used to calculate ground-truth. Using knowledge of the individual targets, accurate centroids for the disks were computed. Flow was calculated from the motion of each centroid for a total of 42 flow vectors. For the third sequence, the scene was ensured to be significantly non-planar.

Tables 3.1 and 3.2 show results. The error metric is the mean angular error in degrees [BFB92]:

$$E_a = \frac{1}{n}\sum_{i=1}^{n} \cos^{-1}(\hat{\mathbf{v}}_{\mathbf{t}_i}^{\mathbf{T}} \hat{\mathbf{v}}_{\mathbf{e}_i})$$

where $\hat{\mathbf{v}}_{\mathbf{t}_i}$ is a unit three-vector in the direction of the true flow and $\hat{\mathbf{v}}_{\mathbf{e}_i}$ is a unit three-vector in the direction of the estimated flow. The multi-scale optical flow algorithm did well for small motions but poorly for large ones. Since the range map for the Yosemite sequence is nearly planar, the difference in performance between the discrete algorithms is less significant than those for the real sequence. It is clear that the rigid algorithm provides the best optical flow estimates.

Unfortunately, optical flow algorithms tested on the Yosemite sequence typically use the version with clouds. [BFBB93] provides results for many algorithms and, of the methods that produce flow at every pixel, the best obtained a mean angular error of 10.44^o. All algorithms presented here surpass that benchmark and it must be presumed that this is due to the difficulty in estimating flow for the clouds. Since the clouds move independently,

sequence →	**Real 1**	**Real 2**
interval →	**1**	**1**
msof	3.28^o	3.63^o
planar	2.87^o	4.67^o
rigid	2.05^o	1.18^o

Table 3.2: Mean angular error in flow vectors for three different algorithms for the GRASP Laboratory sequences. The interval refers to the temporal sampling of the sequence (e.g. every frame was used, every second frame, every third, etc.).

the rigid algorithm can not be tested on this sequence.

3.3 Conclusion

A multi-scale algorithm for estimating optical flow based on an uncalibrated camera moving through a rigid world has been presented. Its implementation is only slightly more complicated and time-consuming than standard multi-scale algorithms. In situations where the camera may be undergoing relatively large motions, the superiority of the rigid model has been demonstrated on both synthetic and real sequences.

However, this algorithm does not resolve the confounding between intrinsic parameters and motion [Oli98]. Also, the trajectory reconstruction is not Euclidean as desired.

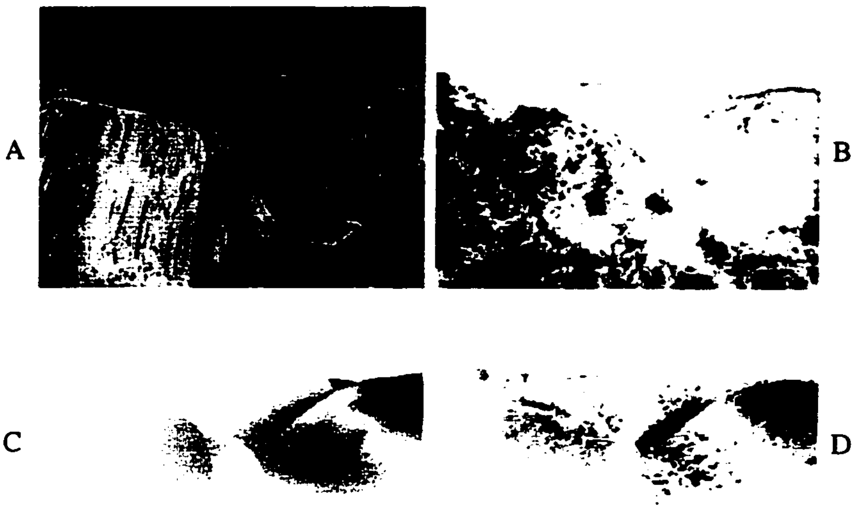


Figure 3.2: Sample image from the Yosemite sequence (A) and the angular error metric of the computed flow fields for the Yosemite sequence by the msof (B), planar (C), and rigid (D) algorithms with a frame interval of four. White corresponds to 0^o of error and black to 45^o.

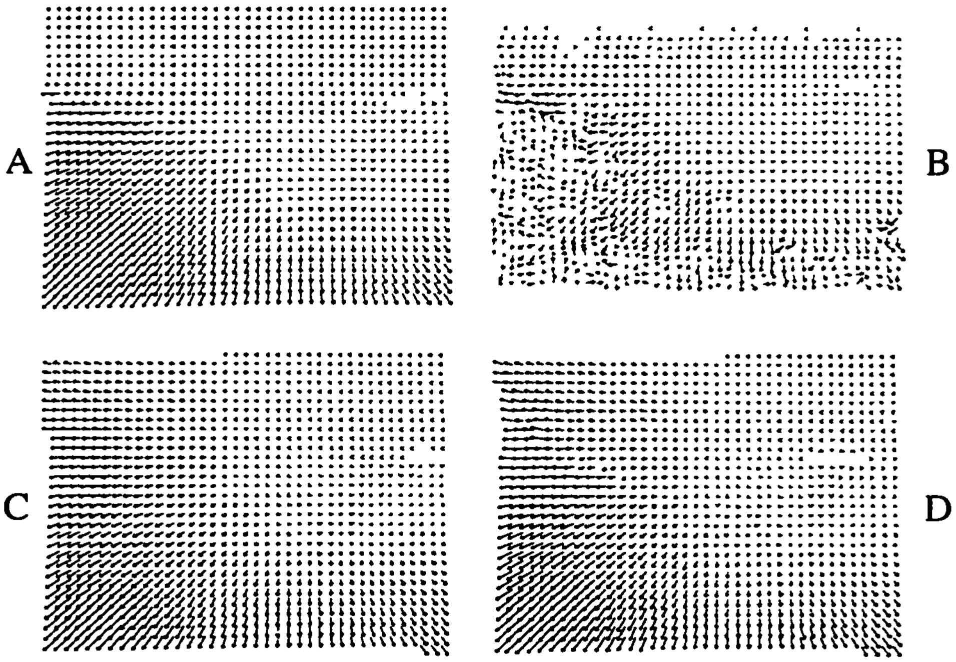


Figure 3.3: True flow field (A) and computed flow fields for the Yosemite sequence by the msof (B), planar (C), and rigid (D) algorithms with a frame interval of four.

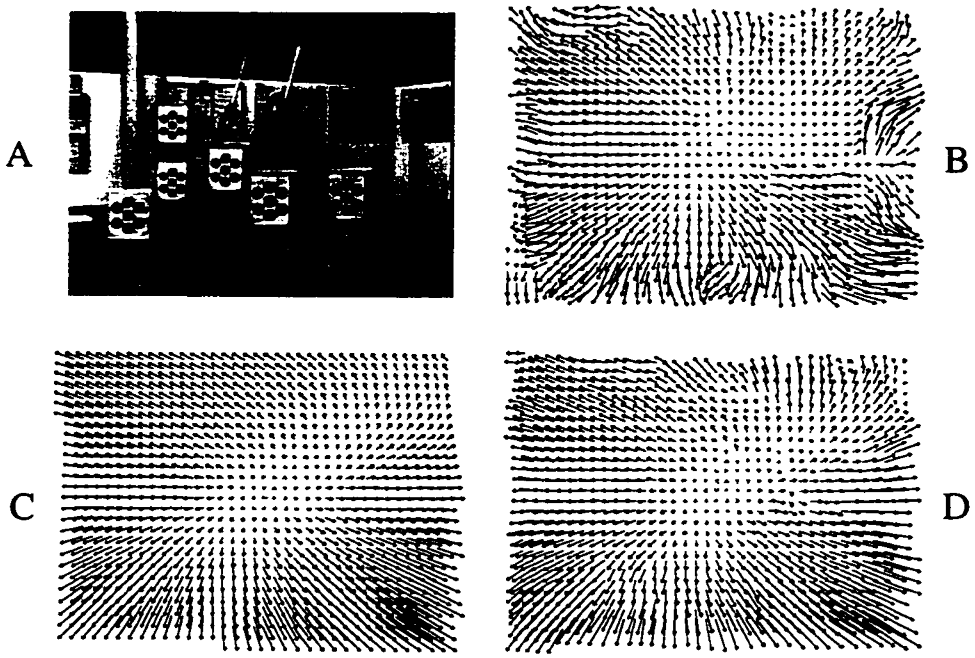


Figure 3.4: Sample image from the real sequence (A) and the computed flow fields for the real sequence by the msof (B), planar (C), and rigid (D) algorithms.

Chapter 4

Constrained Self-Calibration

4.1 Overview

The standard algorithms which might be useful for estimating a camera's intrinsic parameters and trajectory are not sufficiently accurate and not practical for all applications. Proposed here is a technical solution to decrease the sensitivity of self-calibration by placing small, planar, easily identifiable targets of known shape in the environment. Assuming an appropriate ratio of size to distance these targets fix the scale and resolve known ambiguities. The proposed algorithm can be applied in any moving camera application and enables the direct merging of several views into a single 3D-representation such as needed during a stereo reconstruction of a room. For other applications the path of the observer is recovered and can be used for global navigation.

The key benefit to using planar targets is the reduction in degrees of freedom. A target comprised of n line segments has only six degrees of freedom; a rotation and a translation. If these lines were not constrained, each line would have four unique degrees of freedom implying a total of $4n$ degrees of freedom. Requiring five edges to be a planar target reduces the degrees of freedom from twenty to six. In conjunction with the use of targets, other constraints can be added to the system. Some of the constraints investigated here are rigidly linked cameras, target coplanarity, and planar camera motion.

Targets refer not only to specific items placed in the environment but to any object of known shape and size which can be localized in an image. Doors and windows are potential targets. For driving applications, street signs can be used as targets. In multi-agent applications each agent can carry targets, the known shape of each agent could be used, or the agents can drop targets as they travel. These types of targets are generally not found in a density necessary for the proposed algorithm but can be used to augment the introduced target set.

Figure 4.1 provides an overview of the components of the algorithm.

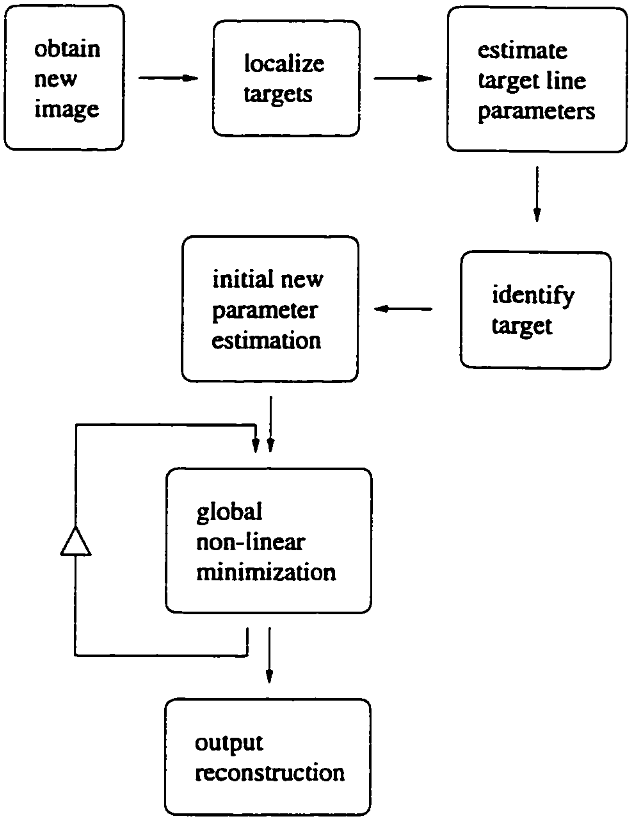

obtain new image
localize targets
estimate target line parameters
identify target
initial new parameter estimation
global non-linear minimization
output reconstruction

Figure 4.1: Algorithm flow chart.

4.2 Target Design and Acquisition

The targets used were designed to be easy to find in an image and were numbered for easy identification. A target is primarily a laser-printed thick-walled black pentagon with a white center (Figure 4.2). The requirements of a white center and five sides provided verification that each dark blot is truly a target. To ensure proper estimation of the orientation of the target, one of the sides of the pentagon was a dark gray instead of black;

this created a bottom to the target.

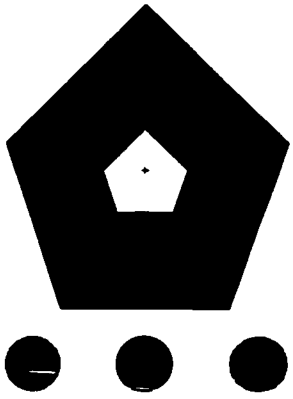


Figure 4.2: Sample target. This target is numbered 100 (see text).

Pseudo-code for the localization and data extraction of the implemented target design is given in Figure 4.3.

```
find dark regions of appropriate area
reject regions without white center of appropriate area
find the corners of each region
reject regions without exactly five corners
reject regions without exactly one grey wall
use the grey wall to order the corners
estimate homography from target to image with corner data
localize edge elements
linearly estimate each edge's data values
compute homography with edge elements
determine target number
```

Figure 4.3: Pseudo-code for target localization, data and number extraction.

Extracting the corners of a black region with white center is non-trivial due to the discretization of the image. The algorithm proceeds by initially assuming every pixel in the dark region is a potential corner. From these potential corners, the first corner is selected by finding the candidate furthest from the candidate furthest from the image origin. The second corner is then selected to be the candidate furthest from the first corner. Then the following is performed until the corner that would be added does not meet the required threshold. Find the candidate which, if a corner, would maximize the resultant perimeter of the polygon. If the increase in perimeter is greater than a specified threshold (2 pixels was used) then make the candidate a corner, remove all candidated within the polygon from future consideration, and search for another corner.

The corners are ordered at all times to follow the counter-clockwise convention of ordering corners of a convex polygon. Once the grey wall is found, a unique ordering is given by requiring the corners to follow the ordering convention and to have the first and second corner be the endpoints of the grey wall's exterior edge segment. Since the ordering is unique, the correspondence between the known target corner coordinates and the observed corner image coordinates is obtained.

The pixels corresponding to the line segments of the pentagon are found by the following algorithm. First, a projective transformation from target coordinates to image coordinates is computed using the localized target corner points. From the projection transformation, the pixels corresponding to the line segment between the black pentagon and the white paper are computed. These are adjusted by a one-dimensional search - roughly perpendicular to the orientation of the line - to the actual edge coordinates. Specifically, the projective transformation provides approximations to the projected corner points. Since the desired edge pixels are components of a continuous line segment, the algorithm starts at one corner and moves towards the other corner one pixel at a time. At each pixel location, the position of the line segment is found in either that row or column depending on the orientation of the line. This is done by performing a one-dimensional search for zero-crossings of the image second-derivative which have a strong first-derivative (arbitrarily required to be, in magnitude, greater than twenty intensity values per pixel) in the area of where the line segment is expected to cross from the preliminary estimation. The second derivative is compute with the seven-tap filter:

$$(0.033589, 0.180366, -0.028225, -0.370850, -0.028225, 0.0180366, 0.033589)$$

and first derivative with the seven-tap filter

$$(-0.008593, -0.115977, -0.240265, 0, 0.240265, 0.115977, 0.008593)$$

(taken from [FS97]). Since the edge elements are searched for in rows or columns, one coordinate of the localized edge point is known exactly. Hence the estimation of slope and intercept of this line segment is a simple linear problem assuming the localized values have a Gaussian distribution about the true value. Also, the covariances of the slope and intercept values are well-defined. For example, a horizontally measured line has the x values without noise and the y values are assumed to have a Gaussian distribution about the true value. The slope m, intercept b, and covariance matrix Λ for these estimates are given by:

$$\begin{aligned} m &= \frac{n\sum_1^n xy - \sum_1^n x \sum_1^n y}{n\sum_1^n x^2 - (\sum_1^n x)^2} \\ b &= \frac{\sum_1^n y - m\sum_1^n x}{n} \\ \Lambda &= \begin{bmatrix} \sum_1^n x^2 & \sum_1^n x \\ \sum_1^n x & n \end{bmatrix}^{-1} \end{aligned}$$

when there are n observations along the line. These line values and covariances, as opposed to corner points, are used during subsequent optimizations for accuracy.

A more conventional representation of the line data is to use a normal vector and the minimal distance of the line to the origin. The primary benefit to this conventional representation is the avoidance of an infinite slope. In the used method, since the orientation of the line is considered before estimating slope, the slope is guaranteed to be at most one. Another benefit to the conventional representation is that error metrics derived from it will use the minimal distance from a point to the observed line as opposed to the 'vertical' distance (Figure 4.4). However, there is a key drawback to the conventional approach. The estimation process to obtain the normal and the distance to the origin is non-linear and hence, while the Cramer-Rao lower bound for the covariances can be determined, the actual covariance matrix for these estimates can not be determined. Without the exact covariance matrix, the error associated with a proposed line segment is not conveniently computable. In the vertical distance representation, the estimation of slope and intercept is linear and the covariance matrix is exact. With only five numbers, the slope, intercept, and three unique values from the symmetric covariance matrix, all the data of the line edge elements is exactly captured.

From the projection transformation - or an improved estimate from the line values - the coordinates of the numbering system are obtained. The numbering system is comprised of five markers where each marker is a white, gray, or black circle representing a one, two, or

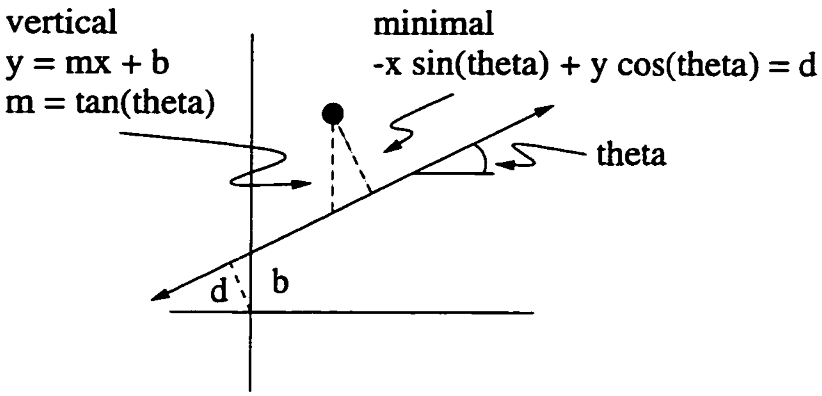


Figure 4.4: Illustration of vertical and minimal distance of a point to a line.

three in a trinary numbering system. In total, this system allows for target numbers from zero to 243.

4.3 Algorithm Metric

Given the line data from each target in each frame and the target labeling, the calibration and transformation parameters are to be found. The Levenberg-Marquardt algorithm is used to minimize a least-squares error metric based on the line data and the projection model for lines. The model is derivable in two steps. First the projection of a point through all the transformations is derived and second the projection of a target line is derived from the point projection equation.

A target point is first transformed into the first frame's coordinate system and then into the current frame's system:

$$\mathbf{x}_f = \mathbf{R}_f \left(\mathbf{R}_t \mathbf{x}_t + \mathbf{t}_t \right) + \mathbf{t}_f$$

For notational simplicity define:

$$\mathbf{R} \equiv \mathbf{R}_f \mathbf{R}_t \qquad \text{and} \qquad \mathbf{t} \equiv \mathbf{R}_f \mathbf{t}_t + \mathbf{t}_f$$

In the case of a stereo configuration the equations for the additional camera are obtained if $\mathbf{R} \equiv \mathbf{R}_s \mathbf{R}_f \mathbf{R}_t$ and $\mathbf{t} \equiv \mathbf{R}_s \mathbf{R}_f \mathbf{t}_t + \mathbf{R}_s \mathbf{t}_f + \mathbf{t}_s$ where the relative transformation between the two cameras is given by $\mathbf{R}_s$ and $\mathbf{t}_s$ (Section 1.4). Given that the target is planar its projection equations can be written as:

$$\mathbf{p} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{C} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ 1 \end{bmatrix} \frac{1}{\begin{bmatrix} r_{31} \; r_{32} \; t_z \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ 1 \end{bmatrix}} = \mathbf{CHx} \frac{1}{\hat{\mathbf{z}}^T \mathbf{H} \mathbf{x}_t}$$

where $\mathbf{C}$ is the matrix of camera intrinsic parameters, r_{ij} is the element of $\mathbf{R}$ in the i-th row and j-th column, and $\mathbf{H}$ is the homography.

Lines in the target are defined by a point $\mathbf{x}_{t_l}$ and a direction vector $\hat{\mathbf{d}}$. A point $\mathbf{x}_t$ of this line is then parameterized by a distance α from $\mathbf{x}_{t_l}$:

$$\mathbf{x}_t = \mathbf{x}_{t_l} + \alpha\hat{\mathbf{d}}$$

After the projection the following relations are obtained:

$$u = fs\frac{\mathbf{h_1^T}\left(\mathbf{x_{t_l}} + \hat{\mathbf{d}}\alpha\right)}{\mathbf{h_3^T}\left(\mathbf{x_{t_l}} + \hat{\mathbf{d}}\alpha\right)} + c_u \qquad v = f\frac{\mathbf{h_2^T}\left(\mathbf{x_{t_l}} + \hat{\mathbf{d}}\alpha\right)}{\mathbf{h_3^T}\left(\mathbf{x_{t_l}} + \hat{\mathbf{d}}\alpha\right)} + c_v \tag{4.1}$$

where:

$$\mathbf{H} \equiv \begin{bmatrix} \mathbf{h_1^T} \\ \mathbf{h_2^T} \\ \mathbf{h_3^T} \end{bmatrix}$$

These equations provide two constraints with one unknown. Using one of the equations to solve for α and substituting into the other equation, a single constraint is obtained. Similarly to the line segment extraction process described in Section 4.2, which equation to substitute into should be determined by the general orientation of the image line so as to ensure the in-image line estimation is a well-posed problem.

As an example, solving for α in the equation for u yields:

$$\alpha = \frac{fs\mathbf{h_1^T x_{t_l}} - (\mathbf{u} - \mathbf{c_u})\mathbf{h_3^T x_{t_l}}}{(u - c_u)\mathbf{h_3^T}\hat{\mathbf{d}} - \mathbf{fsh_1^T}\hat{\mathbf{d}}}$$

Substituting into the equation for v and simplifying results in the constraint for horizontal lines:

$$v = \left[-\frac{1}{s}\frac{\hat{\mathbf{x}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}{\hat{\mathbf{y}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}\right] u + \left[c_v + \frac{c_u}{s}\frac{\hat{\mathbf{x}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}{\hat{\mathbf{y}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})} - f\frac{\hat{\mathbf{z}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}{\hat{\mathbf{y}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}\right]$$

Hence, given the rotations (all $\mathbf{R}_s$, $\mathbf{R}_f$, and $\mathbf{R}_t$), translations (all $\mathbf{t}_s$, $\mathbf{t}_f$, and $\mathbf{t}_t$), the cameras' intrinsic matrices (all $\mathbf{C}$), and the observed orientation of the edge, the expected slope and intercept of a particular line is computable. Since the estimation of the slope and intercept from the pixel data is a linear process the least-squares error metric for the sum of squared vertical distances from the observed edge elements to the expected edge is given by:

$$\begin{bmatrix} m_e - m_o \\ b_e - b_o \end{bmatrix}^T \Lambda^{-1} \begin{bmatrix} m_e - m_o \\ b_e - b_o \end{bmatrix}$$

where m_e is the expected slope, m_o the observed slope, b_e the expected intercept, b_o the observed intercept, and Λ is the covariance matrix of the estimates m_o and b_o. In the case of a horizontal line:

$$\begin{aligned} m_e &= -\frac{1}{s}\frac{\hat{\mathbf{x}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}{\hat{\mathbf{y}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})} \\ b_e &= c_v + \frac{c_u}{s}\frac{\hat{\mathbf{x}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}{\hat{\mathbf{y}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})} - f\frac{\hat{\mathbf{z}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})}{\hat{\mathbf{y}}^T\mathbf{H}^{-T}(\mathbf{x}_{t_l} \times \hat{\mathbf{d}})} \end{aligned}$$

The metric to be mimized is therefore the summation of the errors of each feature of each target in each image.

4.4 Rotation Representation

Of primary importance is the maintenance of the orthonormality of the rotation matrices. The algorithm implements the Rodriguez representation so as to allow only three degrees of freedom for the rotation matrix.

A rotation matrix $\mathbf{R}$ has three degrees of freedom and these parameters are represented by the vector $\mathbf{w}$. The vector $\mathbf{w}$ defines an axis of rotation, $\frac{\mathbf{w}}{|\mathbf{w}|}$, and an angle $|\mathbf{w}|$. The Rodriguez equation relates the rotation matrix to the rotation vector:

$$\mathbf{R} = \mathbf{I}_3 + \frac{\sin|\mathbf{w}|}{|\mathbf{w}|}\mathbf{w}^\times + \frac{1-\cos|\mathbf{w}|}{|\mathbf{w}|^2}\mathbf{w}^\times\mathbf{w}^\times \qquad ,$$

where $\mathbf{I}_3$ is the three-by-three identity matrix, $\mathbf{w} \equiv [w_x \; w_y \; w_z]^T$, and $\mathbf{w}^\times$ is the skew-symmetric matrix representation of the cross-product operation with $\mathbf{w}$:

$$\mathbf{w}^\times \equiv \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} \quad .$$

For use in in the minimization algorithms, the derivative of the rotation matrix with regard to its degrees of freedom is needed. For use later, the component-wise partial derivatives of $\mathbf{w}^\times$ with regard to its components:

$$\frac{\partial \mathbf{w}^\times}{\partial w_x} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \qquad \frac{\partial \mathbf{w}^\times}{\partial w_y} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \qquad \frac{\partial \mathbf{w}^\times}{\partial w_z} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad .$$

Notationally, the component-wise derivative of the any matrix $\mathbf{M}$ to a component $i \in w_x, w_y, w_z$ of the vector $\mathbf{w}$ will be denoted as $\frac{\partial \mathbf{M}}{\partial i}$. Using this notation, the three partial derivatives of $\mathbf{R}$:

$$\begin{aligned} \frac{\partial \mathbf{R}}{\partial i} &= i\frac{\cos|\mathbf{w}|}{|\mathbf{w}|^2}\mathbf{w}^\times - i\frac{\sin|\mathbf{w}|}{|\mathbf{w}|^3}\mathbf{w}^\times + \frac{\sin|\mathbf{w}|}{|\mathbf{w}|}\frac{\partial \mathbf{w}^\times}{\partial i} + i\frac{\sin|\mathbf{w}|}{|\mathbf{w}|^3}\mathbf{w}^\times\mathbf{w}^\times \\ &\quad -2i\frac{1-\cos|\mathbf{w}|}{|\mathbf{w}|^4}\mathbf{w}^\times\mathbf{w}^\times + \frac{1-\cos|\mathbf{w}|}{|\mathbf{w}|^2}\left(\frac{\partial \mathbf{w}^\times}{\partial i}\mathbf{w}^\times + \mathbf{w}^\times\frac{\partial \mathbf{w}^\times}{\partial i}\right) \quad . \end{aligned}$$

A similar derivation can be found in [Fau93].

Note that the rotation representation and these derivatives are well-defined for $|\mathbf{w}| \approx 0$. Specifically, the following limits exist:

$$\begin{aligned} \lim_{|\mathbf{w}|\to 0} \frac{\sin|\mathbf{w}|}{|\mathbf{w}|} &= 1 \ , \\ \lim_{|\mathbf{w}|\to 0} \frac{1-\cos|\mathbf{w}|}{|\mathbf{w}|^2} &= \frac{1}{2} \ , \\ \lim_{|\mathbf{w}|\to 0} \left(\frac{\cos|\mathbf{w}|}{|\mathbf{w}|^2} - \frac{\sin|\mathbf{w}|}{|\mathbf{w}|^3}\right) &= -\frac{1}{3} \ , \\ \lim_{|\mathbf{w}|\to 0} \left(\frac{\sin|\mathbf{w}|}{|\mathbf{w}|^3} - 2\frac{1-\cos|\mathbf{w}|}{|\mathbf{w}|^4}\right) &= -\frac{1}{12} \ . \end{aligned}$$

When computing rotation matrices and derivative of rotation matrices, double precision numbers must be used to ensure reasonable accuracy. To account for small angles of rotation in these computations, when $|\mathbf{w}| \leq 0.001$ the following Taylor approximations are used for the functions:

$$\begin{aligned}
\frac{\sin|\mathbf{w}|}{|\mathbf{w}|} &\approx 1-\frac{|\mathbf{w}|^2}{6}, \\
\frac{1-\cos|\mathbf{w}|}{|\mathbf{w}|^2} &\approx \frac{1}{2}-\frac{|\mathbf{w}|^2}{24}, \\
\left(\frac{\cos|\mathbf{w}|}{|\mathbf{w}|^2}-\frac{\sin|\mathbf{w}|}{|\mathbf{w}|^3}\right) &\approx -\frac{1}{3}+\frac{|\mathbf{w}|^2}{30}, \\
\left(\frac{\sin|\mathbf{w}|}{|\mathbf{w}|^3}-2\frac{1-\cos|\mathbf{w}|}{|\mathbf{w}|^4}\right) &\approx -\frac{1}{12}+\frac{|\mathbf{w}|^2}{180}.
\end{aligned}$$

All of the Taylor approximations for these functions are even functions and hence the approximations are $O(|\mathbf{w}|^4)$.

4.5 Constraints

The primary advantage of using planar targets is the reduction in the number of degrees of freedom. Other constraints are practical in real world applications including:

- rigidly linked cameras (stereo),
- parallel or coplanar targets,
- planar observer motion or motion without a vertical component.

The constraint of having rigidly linked cameras is already incorporated into the described notation with the rotation $\mathbf{R}_s$ and translation $\mathbf{t}_s$. Avoiding this constraint in an algorithm that assumes rigidly linked cameras is not difficult. The observations can be partially described by the position index f and camera index c. The position index describes where the system was when the image was taken; essentially the frame number in the camera's image sequence starting with one. This index provides the mapping from the current position to the world coordinate system by rotation $\mathbf{R}_f$ and translation $\mathbf{t}_f$. The camera index is a labeling for the cameras from one to n where n is the maximum number of cameras in use. This index indicates which intrinsic calibration parameters $\mathbf{C}$ are associated with the camera as well as the position $\mathbf{R}_s$,$\mathbf{t}_s$ of the camera relative to the reference camera. A mapping will be defined so as to modify the indices to obtained the desired result of non-rigidly linked cameras. Converting to a system not constrained by rigidly linked cameras does not affect the intrinsic parameters associated with each camera and hence the camera index c must remain constant. The position index will be mapped to one if f is one and $n(f-1)+c+1$ otherwise.

In many applications, targets will be known to have constraints among their placement. For planar targets, common instances of these constraints occur when targets are placed on walls. Targets on a wall are coplanar and in many instances there will exist walls with a common normal. Hence the constraint that targets share a normal and the constraint of target coplanarity are natural to consider. To constrain target orientations and displacements, rotation $\mathbf{R}_t$ and translation $\mathbf{t}_t$ of each target will be rewritten in terms of two rotations and a translation. The target-to-world coordinate system transformation:

$$\mathbf{x}_w = \mathbf{R}_t\mathbf{x}_t + \mathbf{t}_t$$

is rewritten:

$$\mathbf{x}_w = \mathbf{R}_{xy}(\mathbf{R}_z\mathbf{x}_t + \mathbf{t}_{xyz})$$

where $\mathbf{R}_z$ is a rotation about the z-axis, $\mathbf{R}_{xy}$ is a rotation about an axis in the xy-plane, and $\mathbf{t}_{xyz}$ is a translation vector. Essentially, $\mathbf{R}_t = \mathbf{R}_{xy}\mathbf{R}_z$ and $\mathbf{t}_t = \mathbf{R}_{xy}\mathbf{t}_{xyz}$. To constrain targets to be parallel - to have the same normal - the $\mathbf{R}_{xy}$ of each target are set to depend on the same two minimization parameters. For two targets to be coplanar, the previous orientation constraint is used as well as requiring both targets to share the $\hat{\mathbf{z}}^T\mathbf{t}_{xyz}$ parameter. However, in practice, this last constraint can cause difficulties in numerical minimizations due to the effects of changes in rotation on the translation of the target. To reduce this problem, a minor modification to the representation is used in the minimization:

$$\mathbf{x}_w = \mathbf{R}_{xy}\left(\mathbf{R}_z\mathbf{x}_t + \begin{bmatrix} \hat{\mathbf{x}}^T\mathbf{t}_{xyz} \\ \hat{\mathbf{y}}^T\mathbf{t}_{xyz} \\ 0 \end{bmatrix}\right) + \hat{\mathbf{n}}\hat{\mathbf{z}}^T\mathbf{t}_{xyz}$$

where $\hat{\mathbf{n}} \equiv \mathbf{R}_{xy}\hat{\mathbf{z}}$ and $\hat{\mathbf{n}}$ does not vary as $\mathbf{R}_{xy}$ varies; it is held constant over a minimization iteration. After the minimization iteration, the resultant values are converted back to the correct representation.

Finally, planar motion of the observer will be considered. This constraint can be enforced by requiring for all frames that $\hat{\mathbf{y}}^T\mathbf{t}_f = 0$ and $\mathbf{R}_f\hat{\mathbf{y}} = \hat{\mathbf{y}}$; the rotation is about the y-axis. Note that the coordinate system of the reference camera can no longer be used to define the orientation of the world coordinate system and hence the convention of $\mathbf{R}_s$ being an identity matrix for the reference camera is broken. A lesser motion constraint, that there is no vertical motion of the cameras, is considered as well. This is identical to planar motion except that the rotation axis need not be aligned with the y-axis.

4.6 Algorithm

Pseudo-code for the minimization algorithm is presented in Figure 4.5. Two of the lines are marked optional. These instructions were found to greatly reduce the number of iteration performed by the Levenberg-Marquardt algorithm but did not affect the resultant values of the estimated parameters.

```
initial estimation of new target and frame values
improve target and frame value estimates (optional)
flip planes as necessary
improve orientation estimate of each plane (optional)
non-linear minimization over all parameters
remove targets from seen list which fail observability test
```

Figure 4.5: Pseudo-code for the estimation process.

The overall initialization of the algorithms requires initial values for each camera's intrinsic parameters and position. In all experiments the image center of each camera was set to the physical center of the image, the focal length to the nominal value given by the lens, the scale factor to one, and the initial position of each camera to be hand-measured to a particular camera which was assumed to be at the origin with its optical axis aligned with the z-axis. Substantial variations of these values did not affect the resultant parameters except when the initial relative orientations of the cameras was incorrect by a large amount (e.g. thirty degrees) or when the angle between the true and used camera position was large (e.g. thirty degrees). Placing all the cameras at the origin always worked. Naturally, the accuracy of the initialization values had significant impact on the execution time.

As data is obtained, initial values for the positions of unseen targets and the new position of the observer must be obtained. New targets are first placed by using a simple pose estimation algorithm. This result is then used in a non-linear minimization which takes into account the constraints of orientation and coplanarity which might exist. If the root-mean-squared-error of this estimation is sufficiently small, arbitrarily defined as under three pixels, the target is declared initialized; otherwise initialization of the target will be tried again with the collection of new data. If there is at least one seen target in the new frame then it is initialized to have the parameters of the last frame seen by the same camera. A non-linear improvement is made to these values subject to any motion constraints that might be in use.

Since the targets are planar, there generally exists two minima to the orientation of the target. Given one minima - which defines the relative position of the target to the camera - and the camera's intrinsics the other minima is found by taking the mirror image of the target about the plane containing the relative translation of the target and the vector perpendicular to this translation and the normal of the target. Unfortunately, the initial pose is not exactly the minima so after the mirror image is obtained a non-linear refinement must be performed. Also, for execution time considerations, only the translation vector relevant to the first sighting of the target is considered. The residual of the entire estimation is considered at the two minima and the result corresponding to the lower residual is kept. When orientation constraints are in effect, instead of potentially flipping each target individually the test is performed on all targets with the same orientation by comparing current orientation and the flipped orientation of an arbitrary target (e.g. the first seen target in the list specifying the linked orientations).

Another ambiguity that can occur in the estimation process is when a target is believed to be behind the camera; the bas-relief ambiguity. Clearly, a target behind the camera is out of its field-of-view and could not have been imaged. Every target is tested to verify it is in front of every camera that imaged the target in every frame the target was seen. Failure to pass this test marks the target uninitialized and the target is therefore considered a new target in the next invocation of the algorithm.

4.7 Experimental Results

4.7.1 GRASP Laboratory Sequence

A seventeen frame image sequence was taken at the GRASP lab to demonstrate the algorithm. The visual system consisted of two cameras rigidly mounted to a horizontal link that moved through the scene so as to maintain a table in the right camera's field of view. The scene consisted primarily of a table and two walls on which planar targets - the pentagonal objects - were hung. Figure 4.6 provides a sample stereo pair to depict the experimental setup.

In total eight targets were imaged by the cameras. Seven of these were visible in the first stereo pair and, thereafter, typically three or four of the targets were clearly visible. The eight target was only seen twice; in the tenth and seventeenth frame by the left camera. The displacement of the targets from the corner ranged roughly up to 2.25 meters on the left wall and 2 meters on the right. The targets are 19.5 centimeters tall.

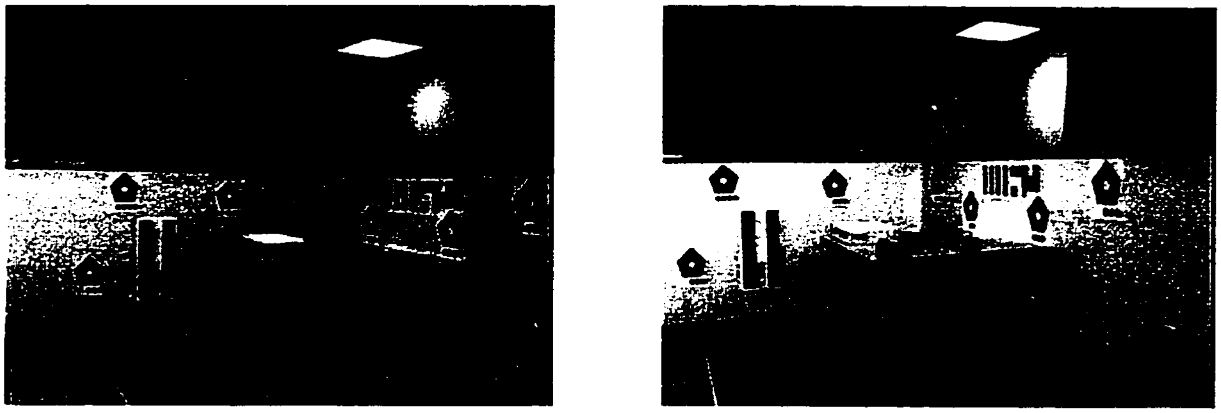


Figure 4.6: Sample stereo images from the GRASP lab sequence. The pentagonal objects are the targets for the algorithm.

The first results are a set of augmented reality images shown in Fig. 4.7. To create these, the dimensions of the table were measured relative to a coordinate system defined by the two targets 'on' the table. The line through both targets' origins was chosen as the x-axis and the average of their normals as the z-axis. The system's origin coincides with the left target's origin. Given these relative measurements, the reconstruction provides a mapping into the coordinate system of each camera and hence allows for projection.

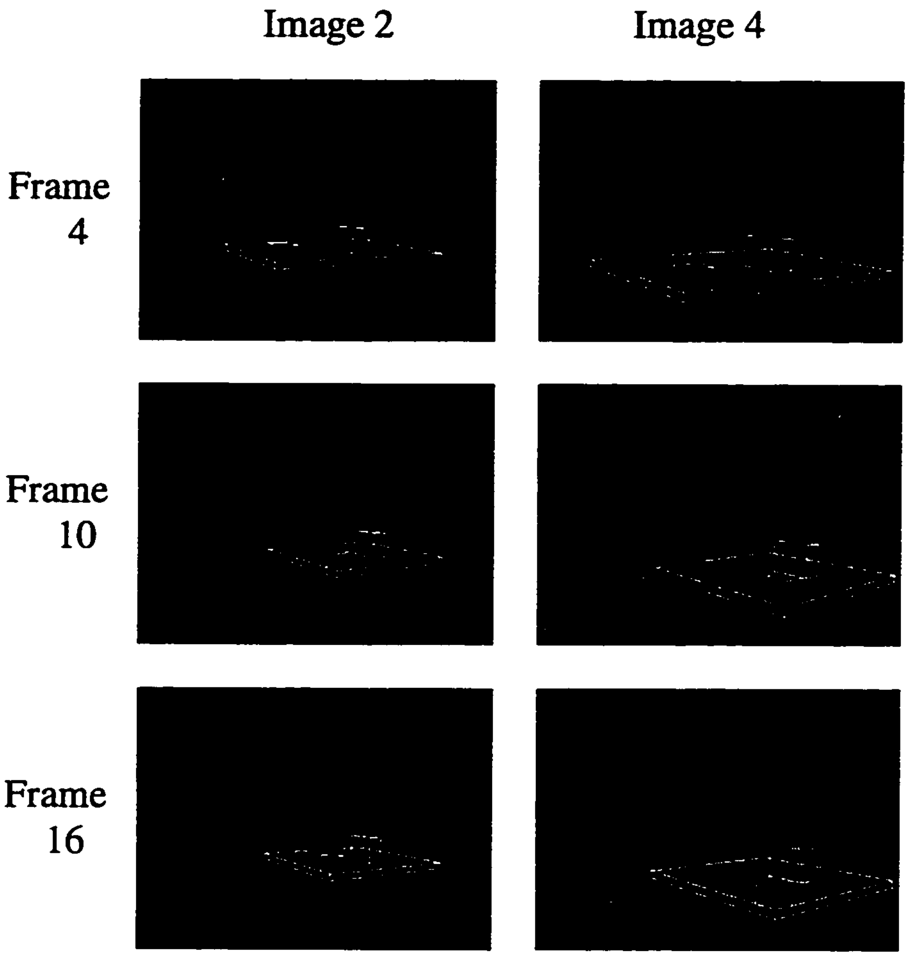


Figure 4.7: On the left are the results for image number two; on the right for image four. The amount of processed frames spans vertically and is shown for after the fourth, tenth, and sixteenth frames. All images are from the left camera.

Two boxes were also projected into the images. The initially huge errors present in the projection are due to the implausibility of separating calibration and pose values from the small number of seen targets and the position they occupy; although not presented, the reprojection errors for the targets is *smallest* during these frames since the ratio of degrees of freedom to observed data is highest.

Another qualitative result is given by the bird's eye view of the reconstruction in Fig. 4.8. This shows the progression of reconstruction values over the frame number (time). The T's in the image are the targets and the robot and its trajectory are also shown. The targets were hung in a corner so the true configuration has two lines that are perpendicular; as roughly shown by the seventeenth frame in the figure. The improvement in the reconstruction is clearly evident by looking at the angle created by the two sets of targets. Of key note is the misalignment of the leftmost target in frames numbered ten and thirteen. This target appears in the sequence in only two images; the left frame of the tenth and seventeenth. Initially, the pose is incorrect but the second viewing provided enough information to correct the target's orientation.

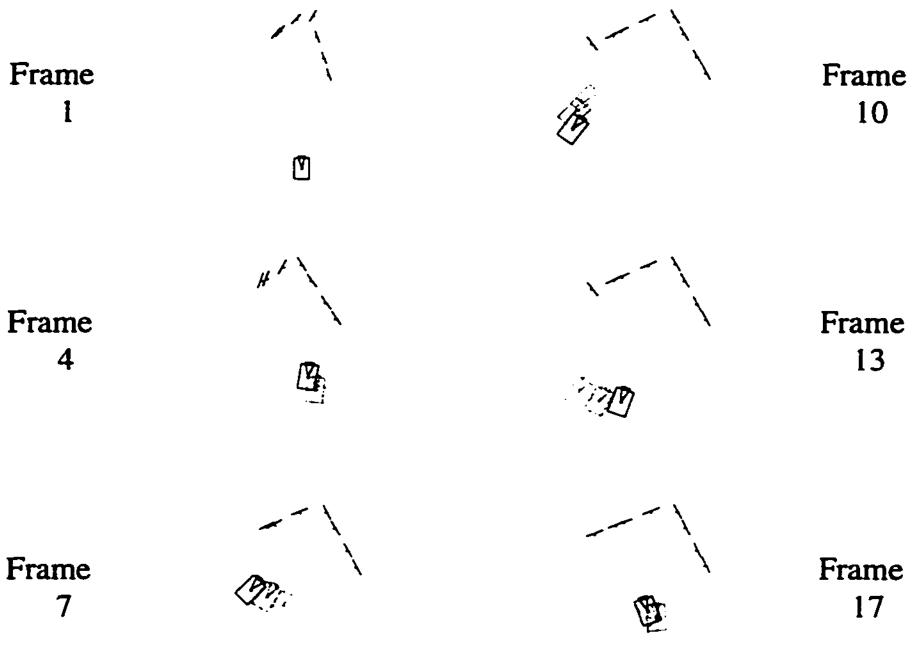


Figure 4.8: Bird's eye view of the reconstruction. The targets are represented with T's and the robot with a schematic. To show trajectory, past positions of the robot are included.

Quantitatively, two results are provided. The first is a comparison of the estimated separation between the cameras of the stereo pair. The estimate value versus frame number is displayed in Fig. 4.9; the nominal value was measured to be 41.75 centimeters. Of course, the true separation is unmeasurable so, as a proxy, the distance was measured between the respective corners of the cameras' cases. The second result, also shown in Fig. 4.9, plots the coplanarity error over the frame number. For this graph the misaligned target previously discussed was not included and the values for each wall were computed separately and appropriately averaged. The metric is defined as:

$$\frac{1}{n(n-1)}\sum_{i}\sum_{j\neq i}\left|\hat{\mathbf{z}}^T\mathbf{R}_i^T\left(\mathbf{t}_j-\mathbf{t}_i\right)\right| \qquad \text{where } i,\, j \text{ range from 1 to } n$$

Essentially, this computes an average distance from each target's plane to every other target's origin. If the targets were perfectly coplanar the result would be zero.

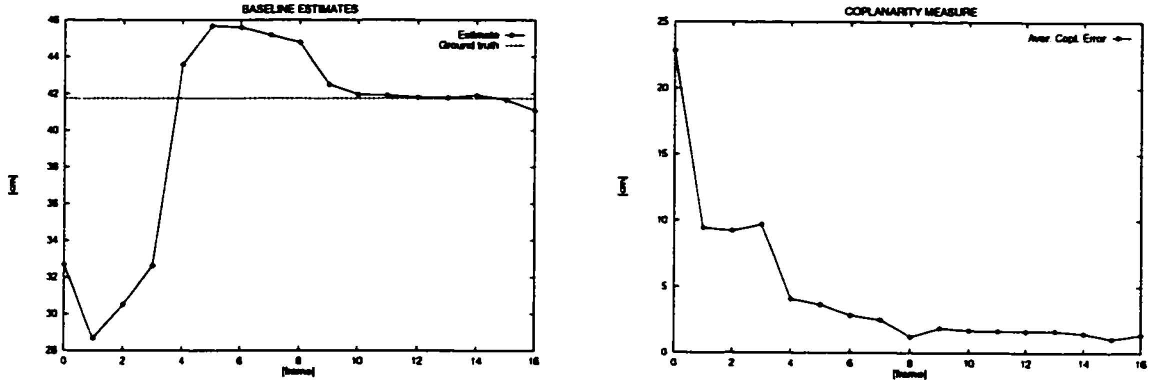


Figure 4.9: On the left are estimated baseline lengths during the reconstruction; the nominal value is displayed as well. On the right is the coplanarity error metric plotted over frame number excluding the mis-oriented target.

4.7.2 100 Calibration-Length Sequences

The testbed for this experiment is a set of one hundred image sequences. A sequence is comprised of two images (a stereo pair) taken at each of nine positions. The general motion of the robot is given by Figure 4.10. Position 1 was roughly 1.5 meters from the corner and Position 2 roughly four meters from the corner.

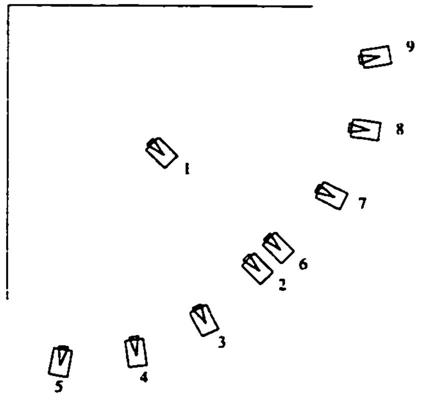


Figure 4.10: Robot imaging positions.

The targets were placed on the two walls drawn as line segments in Figure 4.10. Figure 4.11 depicts the layout of the targets and indicates which distances were measured to be used in evaluating the reconstruction.

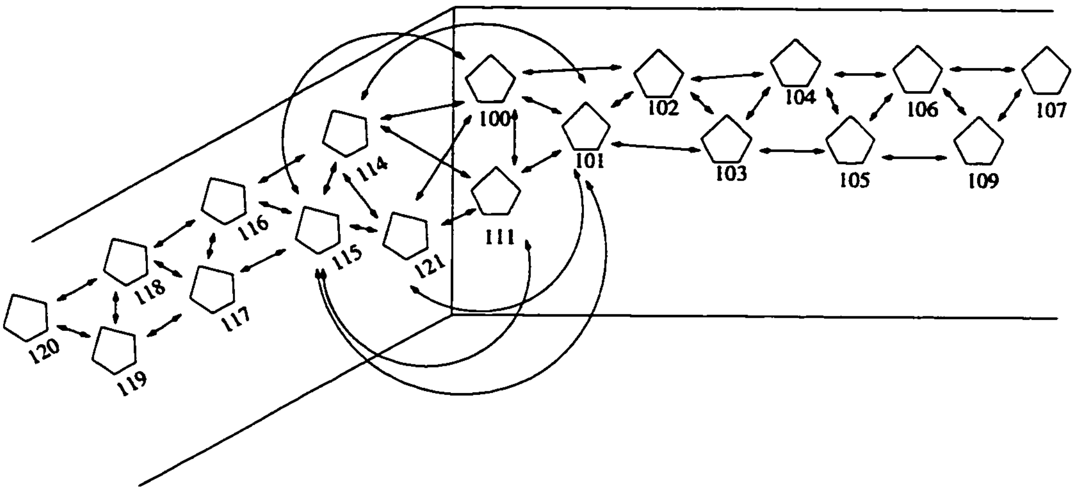


Figure 4.11: Depiction of target positions (not to scale). The connecting lines represent measured distances. The numbering of each target is indicated as well.

From each image, two data sets are considered: the full nine positions and the subset comprised of the first, third, fifth, seventh, and ninth positions. For each data set eighteen estimations are performed resulting in thirty-six measurements. The eighteen estimations are formed by combinations of possible constraints on targets (none, orientation specified, coplanarity), whether or not the stereo configuration was enforced, and possible constraints on estimated motion (none, no vertical motion, planar motion).

The robot used is a Nomadic Technologies Incorporated XR4000. This robot is capable of holomonic motion in the plane of the floor. However, the cameras are mounted on the robot's exterior shell which can be seen to slightly shift with regard to the robot's base as the robot moves. Also, the GRASP Laboratories floor, as tested by a level, is known to have different gradients at different positions. Given that the effect of a small rotation of the robot system produces a large motion in the image, it is expected that the planar motion constraint will be very inaccurate to the actual motion. This is observed by comparing Table 4.1 with Table 4.2. Table 4.1 presents the average and standard deviation of the residual root-mean-squared-error (RMSE) over the one hundred sequences for the trials with the planar motion constraint. Table 4.2 presents the same data for the other trials. Looking at the sequences of length nine, the planar motion constraint increases the residual by a factor of three. If the success of a sequence is arbitrarily defined as the minimization resulting in a RMSE of less than 1.5 pixels, then clearly the planar constraint is inconsistent

seq. length	target constr.	stereo constr.	mean RMSE in pixels	std. RMSE in pixels	success mean RMSE in pixels	success std. RMSE in pixels	num. of successes
5	none	none	1.257	0.908	0.966	0.033	23
5	none	stereo	1.078	0.083	0.957	0.036	14
5	ori.	none	1.207	1.300	0.933	0.039	80
5	ori.	stereo	0.983	0.073	0.944	0.035	65
5	planar	none	1.791	3.179	0.911	0.090	80
5	planar	stereo	0.988	0.139	0.950	0.036	76
9	none	none	1.706	0.085	n/a	n/a	0
9	none	stereo	1.765	0.068	n/a	n/a	0
9	ori.	none	1.700	0.044	n/a	n/a	0
9	ori.	stereo	1.781	0.049	n/a	n/a	0
9	planar	none	2.556	3.346	n/a	n/a	0
9	planar	stereo	1.839	0.207	n/a	n/a	0

Table 4.1: Residual of estimation assuming planar motion. A success is defined as a RMSE under 1.5 pixels.

with the physical process that produced the sequence since not a single sequence had a residual less than this value. The selection of this threshold is guided by examining the observed cumulative probability density functions of the estimation residuals under the different constraint scenarios. These CDF's are presented in Appendix A.

These residual tables do have some unexpected results. Adding constraints to a minimization can not decrease the resultant residual; the unconstrained problem can always assume the constrained value to obtain the same residual. However, in almost every situation of adding or strengthening a constraint the residual decreases. Also, as the sequence length increases the residual is expected to increase since the ratio of degrees of freedom to observations decrease as sequence length increases. This implies the minimization is somewhat ill-posed; there are local minima or at least very shallow regions of decent. To verify this, for a few situations the solution values to a more constrained problem were used to initialize the minimization of a less constrained problem. The resultant residual was always lower than the constrained problem residual. Using the estimated parameters from the longer sequence in the five frame sequence produced similar results.

Table 4.3 presents statistics for the estimated separation of the cameras. The nominal value is 160 millimeters and this was obtained by measuring from a corner of the first

seq.	target	stereo	motion			success	success	num. of
length	constr.	constr.	constr.	mean	std.	mean	std.	successes
				RMSE	RMSE	RMSE	RMSE	
				in pixels	in pixels	in pixels	in pixels	
5	none	none	none	1.746	3.856	0.501	0.090	85
5	none	none	no vert.	0.822	1.450	0.500	0.078	94
5	none	stereo	none	0.649	0.576	0.571	0.155	98
5	none	stereo	no vert.	0.527	0.105	0.527	0.105	100
5	ori.	none	none	0.655	1.233	0.353	0.021	94
5	ori.	none	no vert.	0.520	0.828	0.395	0.091	97
5	ori.	stereo	none	0.375	0.024	0.375	0.024	100
5	ori.	stereo	no vert.	0.521	0.957	0.376	0.044	97
5	planar	none	none	1.101	2.487	0.396	0.051	89
5	planar	none	no vert.	2.865	6.661	0.428	0.103	78
5	planar	stereo	none	0.466	0.392	0.401	0.028	96
5	planar	stereo	no vert.	0.769	1.507	0.425	0.057	91
9	none	none	none	0.779	2.102	0.556	0.067	97
9	none	none	no vert.	0.549	0.059	0.549	0.059	100
9	none	stereo	none	0.616	0.115	0.601	0.078	97
9	none	stereo	no vert.	0.586	0.082	0.580	0.061	99
9	ori.	none	none	0.405	0.018	0.405	0.018	100
9	ori.	none	no vert.	0.409	0.013	0.409	0.013	100
9	ori.	stereo	none	0.425	0.015	0.425	0.015	100
9	ori.	stereo	no vert.	0.428	0.014	0.428	0.014	100
9	planar	none	none	0.468	0.153	0.447	0.017	98
9	planar	none	no vert.	0.492	0.196	0.464	0.020	98
9	planar	stereo	none	0.493	0.143	0.479	0.022	99
9	planar	stereo	no vert.	0.514	0.132	0.501	0.029	99

Table 4.2: Residual of estimation. A success is defined as a RMSE under 1.5 pixels.

camera to the corresponding corner of the second camera. Table 4.4 provides data on the accuracy of the reconstruction of the distances depicted in Figure 4.11. Since the two tables present results in which the constraints do match the reality of how the sequences were collected, it is expected and demonstrated that the incorporated constraints improve the accuracy of the results. For the distance table, a requirement that the residuals of both targets involved in the distance measurement have residual less than 1.5 pixels to be included in the presented data. This is not a hard requirement to meet as shown in Table 4.2. In fact, this requirement was observed to only be violated in the case of an obvious outlier in the result; a failure of the estimation algorithm.

Analysis of the baseline and distance results shows that the use of more data, a switch from length five to length nine sequences, tends to improve the estimates as well as reduce the variance in the estimates. However, this improvement is not dramatic implying that the amount of data in the length five sequences adequately constrains the results. The incorporation of a stereo constraint offers dramatic improvements in the baseline estimates due to the massive reduction in the associated degrees of freedom. For the distance results, constraints on the placement of the targets limit the relative effectiveness of the stereo constraint. Inversely, the constraints on the target placement had little effect on the baseline estimates while providing significant improvement in the distance results which is where the associated degrees of freedom reside. Overall, the expected result of improved accuracy with decreased degrees of freedom is observed since the constraints match the reality of how the sequences were collected.

seq. length	target constr.	stereo constr.	motion constr.	mean in mm	std. in mm	min. in mm	max. in mm	num. of successes
5	none	none	none	147.460	75.809	56.025	606.169	85
5	none	none	no vert.	152.672	73.798	81.208	607.458	94
5	none	stereo	none	169.702	11.180	127.937	199.493	98
5	none	stereo	no vert.	163.936	6.741	128.646	180.296	100
5	ori.	none	none	159.906	20.375	141.342	295.861	94
5	ori.	none	no vert.	162.932	22.925	137.493	290.635	97
5	ori.	stereo	none	159.349	2.470	146.394	162.663	100
5	ori.	stereo	no vert.	158.948	8.559	149.793	239.596	98
5	planar	none	none	170.565	33.428	118.969	386.486	91
5	planar	none	no vert.	229.300	394.683	122.046	3447.428	80
5	planar	stereo	none	161.747	25.208	153.628	383.863	99
5	planar	stereo	no vert.	155.397	9.766	149.739	241.871	92
9	none	none	none	139.966	34.403	107.211	435.443	99
9	none	none	no vert.	136.777	33.881	105.828	428.998	100
9	none	stereo	none	163.364	5.060	149.154	179.850	100
9	none	stereo	no vert.	161.241	4.425	146.587	188.569	100
9	ori.	none	none	162.189	15.402	142.484	216.353	100
9	ori.	none	no vert.	161.448	12.751	145.677	219.587	100
9	ori.	stereo	none	158.455	2.661	142.241	160.887	100
9	ori.	stereo	no vert.	158.515	2.695	142.243	161.237	100
9	planar	none	none	179.543	19.035	160.756	311.909	98
9	planar	none	no vert.	180.483	20.177	161.597	332.988	98
9	planar	stereo	none	158.950	3.534	155.839	179.890	99
9	planar	stereo	no vert.	159.923	3.864	156.892	183.816	99

Table 4.3: Estimates of baseline length. Nominal value is 160 mm. A success is defined as a RMSE under 1.5 pixels.

seq. length	target constr.	stereo constr.	motion constr.	RMSE in mm	num. of successes
5	none	none	none	141.814	3298
5	none	none	no vert.	164.751	3658
5	none	stereo	none	53.141	3755
5	none	stereo	no vert.	48.003	3891
5	ori.	none	none	37.792	3666
5	ori.	none	no vert.	38.538	3737
5	ori.	stereo	none	39.365	3900
5	ori.	stereo	no vert.	29.852	3801
5	planar	none	none	13.311	3492
5	planar	none	no vert.	15.785	3012
5	planar	stereo	none	13.170	3778
5	planar	stereo	no vert.	15.798	3544
9	none	none	none	101.276	3836
9	none	none	no vert.	79.570	3900
9	none	stereo	none	55.824	3888
9	none	stereo	no vert.	54.751	3888
9	ori.	none	none	55.900	3900
9	ori.	none	no vert.	57.836	3900
9	ori.	stereo	none	48.279	3900
9	ori.	stereo	no vert.	51.360	3900
9	planar	none	none	10.248	3822
9	planar	none	no vert.	10.590	3822
9	planar	stereo	none	9.842	3861
9	planar	stereo	no vert.	10.852	3861

Table 4.4: RMSE of all distances. A success is defined as a RMSE under 1.5 pixels for the sequence as well as the RMSE of each of the two targets for which the distance is being measured having a RMSE under 1.5 pixels.

4.8 Conclusion

A technical solution to decrease the sensitivity of self-calibration by placing small, planar, easily identifiable targets of known shape in the environment was presented. These targets reduce the degrees of freedom of the reconstruction problem and also resolve known ambiguities of self-calibration. The flexibility of the placement of the fiducials implies the need for the targets does not affect the feasibility of using this algorithm.

Similarly to self-calibration techniques, the accuracy of the algorithm suffers when there is insufficient data to prevent ambiguities in the reconstruction. This is demonstrated by the effect of sequence length and various constraints on the accuracy of reconstruction. A key contribution of this work is the progress towards discovering the minimal amount of reconstruction error in self-calibration using current technology cameras. The constrained algorithms have proved to be well-posed - adding constraints improved accuracy - and hence the analysis of accuracy for these algorithms is applicable to less constrained algorithms such as standard self-calibration techniques.

Overall, the algorithm is sufficiently accurate and reliable to be used in applications where the placement of fiducials is acceptable.

Chapter 5

Conclusion

This dissertation explores the problem of extracting camera calibration and trajectory information from a sequence of images. The primary difficulties of this problem stem from the confounding of the intrinsic and motion parameters by the projection process.

A standard solution technique to this problem is off-line camera calibration to obtain the camera intrinsics followed by pose estimation in each frame to estimate the trajectory. Primary concerns with using this technique stem from the impracticality of using a single target in a large workspace and the unreliability of camera calibration from one frame. The first issue is partially addressed by projection approximation algorithms and simple modifications to these algorithms were presented that dramatically improved the accuracy and reliability of these algorithms without substantial sacrifices in execution time. The second issue can be addressed by using multiple viewpoints in the estimation process. However, this is outside the scope of standard calibration algorithms.

Another potential solution is the use of self-calibration algorithms. These methods use a set of correspondences over a set of images to estimate the camera's intrinsic parameters and, potentially, the trajectory of the camera. A key difficulty is finding the correspondences and a natural solution to this problem is the use of optical flow. However, optical flow estimation is typically inaccurate and unusable as a data source for other algorithms. To overcome this an algorithm for estimating optical flow constrained to a rigid motion was developed. This algorithm obtained very accurate flow estimations. The accuracy of the recovered weak-calibration parameters was poor. Also, similar to all self-calibration algorithms, the recovered trajectory must have at least a scale ambiguity which may prevent the techniques from being useful.

The problems of finding data correspondences, defining the metric reference, and removing the scale ambiguity have a simple solution. Small, planar, easily identifiable targets of known shape were placed in the environment. These targets were numbered to solve the correspondence problem. Since the targets are of known dimensions, they create an absolute metric and resolve known ambiguities of self-calibration. Also, they reduce the degrees of freedom of the reconstruction problem and this allows for accurate estimations. Other constraints, involving target placement and observer motion, were explored as well. Overall, the algorithm proved sufficiently accurate and reliably to be used in many applications. The presented constrained self-calibration algorithm can also be viewed as a camera calibration algorithm involving multiple viewpoints and multiple targets.

The observed accuracy and reliability of different components of the reconstruction varied. The values obtained for the camera intrinsics, and the baseline in stereo experiments, were found to be repeatable over many different setups and trajectories. Coupled with the low estimation residual, it is implied that these values were accurate. Target relative pose was also found to be estimated reliable. However, the position of the observer relative to the world coordinate system was not recovered to nearly the same accuracy as the previously mentioned values. These results are intuitively correct. The amount of data relying on, and hence constraining the values for, the camera intrinsics, and baseline in stereo experiments, is very large. The supporting base for the relative position of targets is large. However, the observer pose is only constrained by the observations in a few images and can not be obtained as accurately as the other values.

Analysis of the presented baseline and distance results shows that the use of more data, a switch from length five to length nine sequences, improves the estimates as well as reducing the variance in the estimates. The incorporation of a stereo constraint offers dramatic improvements in the baseline estimates due to the massive reduction in the associated degrees of freedom. For the distance results, constraints on the placement of the targets limit the relative effectiveness of the stereo constraint. Inversely, the constraints on the target placement had little effect on the baseline estimates while providing significant improvement in the distance results which is where the associated degrees of freedom reside. Overall, the expected result of improved accuracy with decreased degrees of freedom is observed since the constraints match the reality of how the sequences were collected.

A large area for future work in constrained self-calibration is in sensitivity analysis. The experimental results presented in this dissertation provides data points for the effect of certain constraints on reconstruction accuracy for a particular experimental setup. However, the effects of these constraints on arbitrary problems remain unexplored. Knowledge of how target density, imaging positions, target placement constraints, and observer motion constraints affect the estimation accuracy will allow for path planning that ensures reliable estimates in minimal execution time. Specifically, the sensitivity to the following factors should be investigated:

- distance from cameras to targets,
- all targets being coplanar,
- all targets being parallel,
- and the amount of target overlap between images.

Techniques for reducing the execution time of the algorithm should be explored. A recursive version of the algorithm, assuming this non-linear problem can be accurately solved recursively, would be of extreme value. A version of the algorithm that reduces the data set by combining well-known targets into one large target has potential assuming the selection of when to combine targets can be automated.

Other constraints which should be analyzed include fixating cameras, pure rotational or translational observer motion, and coplanar lines as opposed to targets. The first two constraints are standard constraints applied in the computer vision literature and their effect on the self-calibration problem should be experimentally explored to determine similarities of self-calibration to these other vision tasks. The final constraint is actually a relaxation of the requirement for having targets. By assuming the lines of a target are only coplanar, as opposed to being in known configuration, intuitions as to the effect of using a target can be obtained. Since the eventual goal is to perform self-calibration without targets, this is clearly an important step.

Appendix A

Residual CDF's

During the experiment many outliers in the resultant data were found. These outliers greatly affect reported means and sample variances and render these values meaningless for analysis of the algorithm's performance. However, to manually remove these outliers would also skew the results and, for real applications, the agent would have to make the determination of outlier versus real data without the knowledge of the approximate true values. As such this decision should be based on a value intrinsic to the estimation process such as the residual of the estimation.

The following figures provide the observed cumulative probability distribution functions for the experiment described in Section 4.7.2. The CDF plots show the ratio of the data that has a residual below the x-axis value. For instance, in Figure A.1 the top left graph shows that 80% of the observations had a residual below roughly 0.75 pixels. A useful feature of this type of graph is the ability to determine the size of the range containing the majority of the observations as well as how many observations were in that range. In the previously referenced graph, about 75% of the observations had a residual of 0.4 pixels to 0.5 pixels. A primary use of these CDF's is to determine a reasonable threshold value in the estimation residual to determine whether the results of the estimation should be considered correct or an outlier. Also, while not mathematically pure, comparing the range from one graph to that of another graph provides a sense of how well the conditions in the graphs relatively apply to the problem being examined.

With the exception of the CDF's for planar motion, requiring the residual to be less than 1.5 pixels implies 78% of the observation would be considered non-outlier in the worst scenario; typically this test passes over 95% of the observations. As such, this threshold is used in reporting the results of Section 4.7.2.

The planar motion constraint is believed to be inappropriate for the experiment (see Section 4.7.2) and this is evident by comparing Figure A.6 with the other two CDF's for length nine sequences. In all cases, the vast majority of the results have an estimation residual under 0.5 pixels in Figure A.4 and Figure A.5 while in Figure A.6 nearly all results occur with a residual over 1.5 pixels implying the planar motion constraint is inconsistent with the motion of the observer during the sequence acquisition. A similar result is found when comparing Figure A.3 with Figure A.1 and Figure A.2.

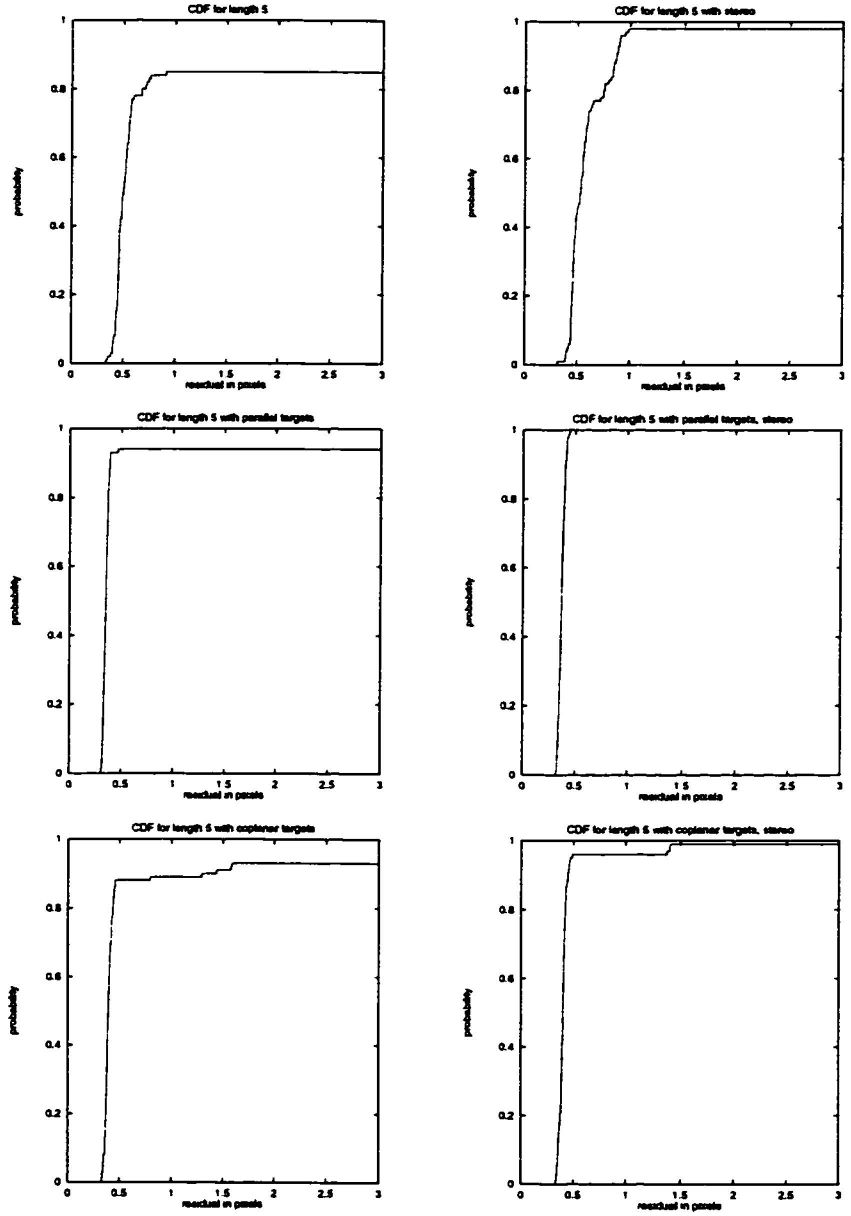
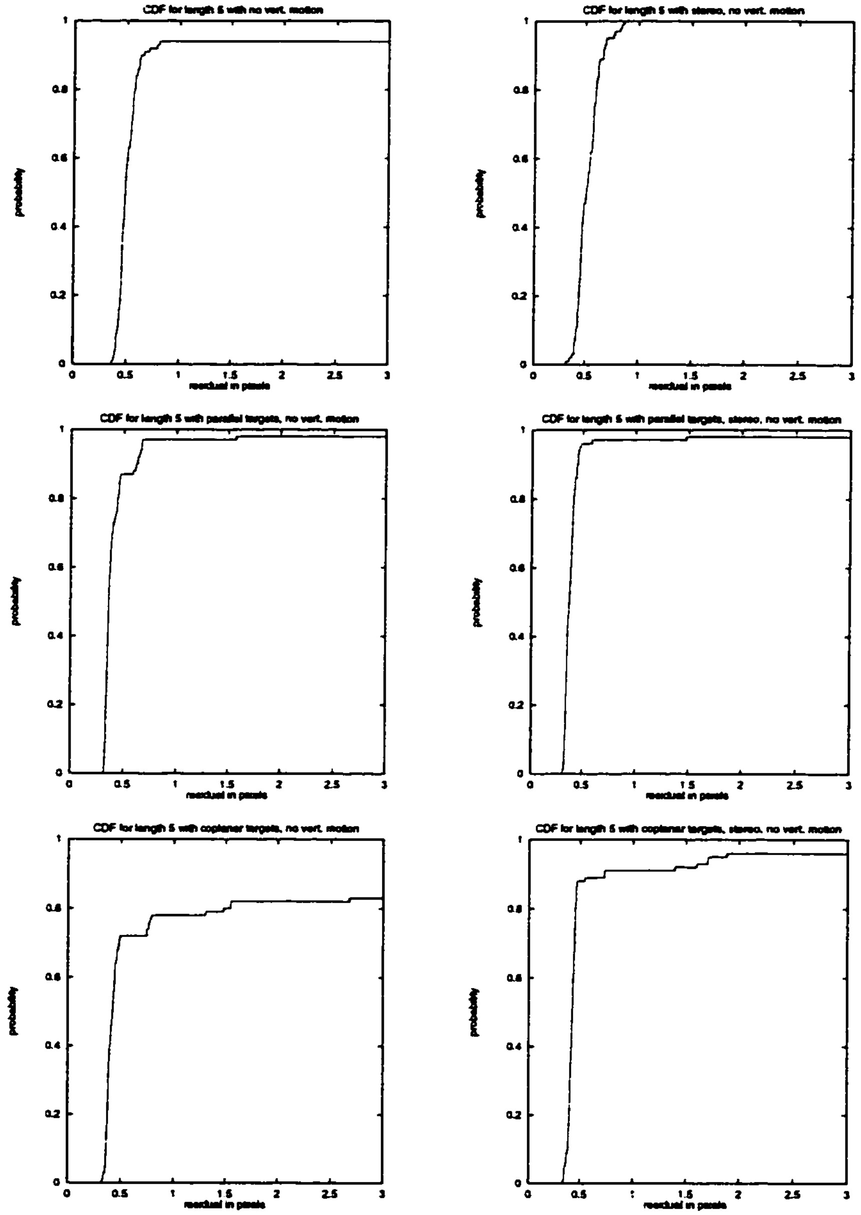


Figure A.1: CDF's for sequences of length five and no motion constraint.



CDF for length 5 with no vert. motion

CDF for length 5 with stereo, no vert. motion

CDF for length 5 with parallel targets, no vert. motion

CDF for length 5 with parallel targets, stereo, no vert. motion

CDF for length 5 with coplanar targets, no vert. motion

CDF for length 5 with coplanar targets, stereo, no vert. motion

Figure A.2: CDF's for sequences of length five and no vertical motion.

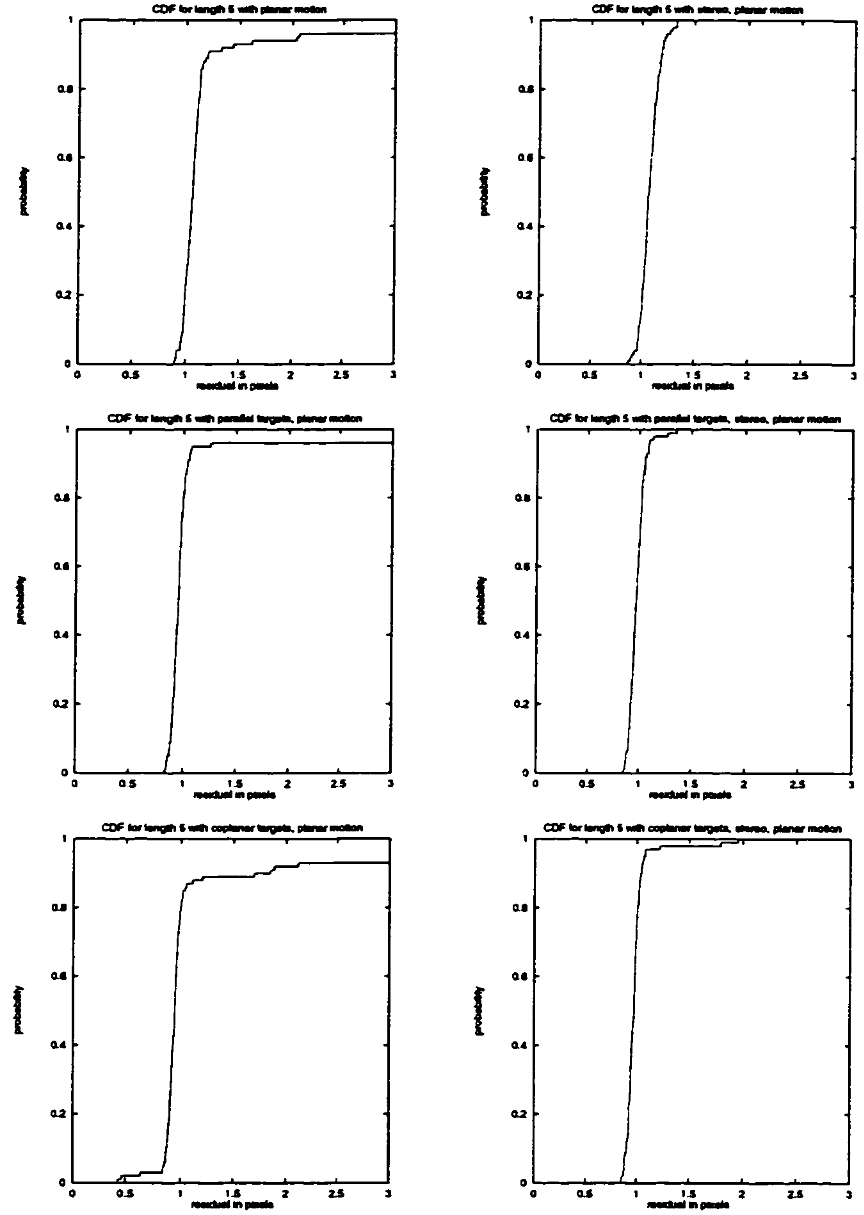


Figure A.3: CDF's for sequences of length five and planar motion.

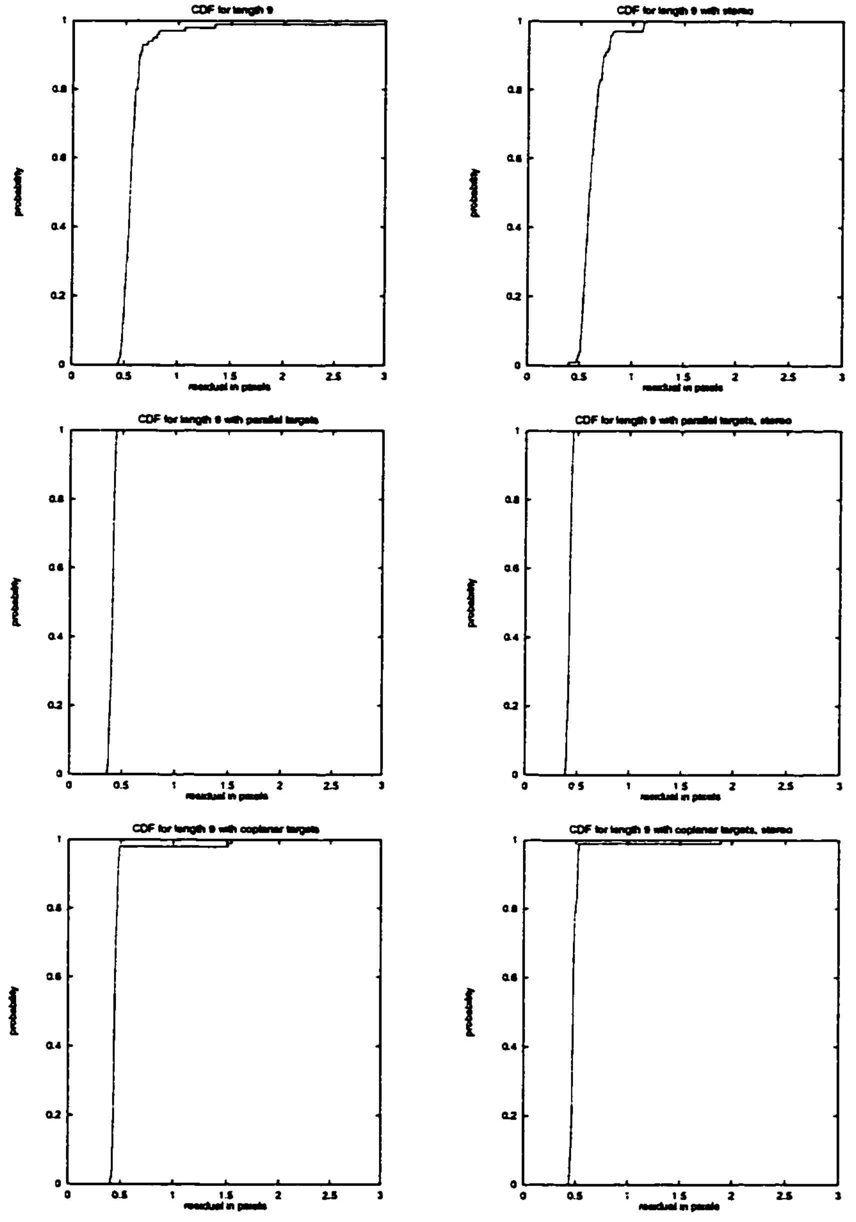


Figure A.4: CDF's for sequences of length nine and no motion constraint.

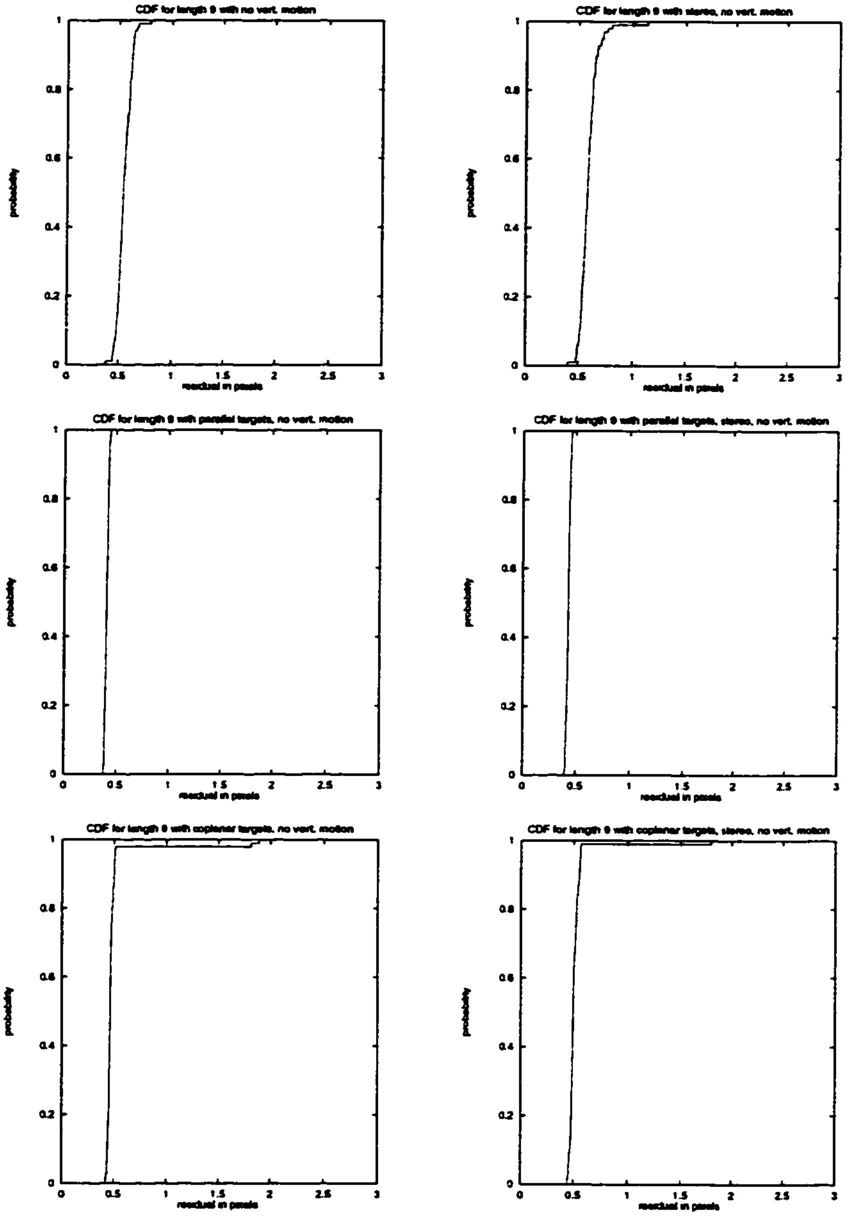


Figure A.5: CDF's for sequences of length nine and no vertical motion.

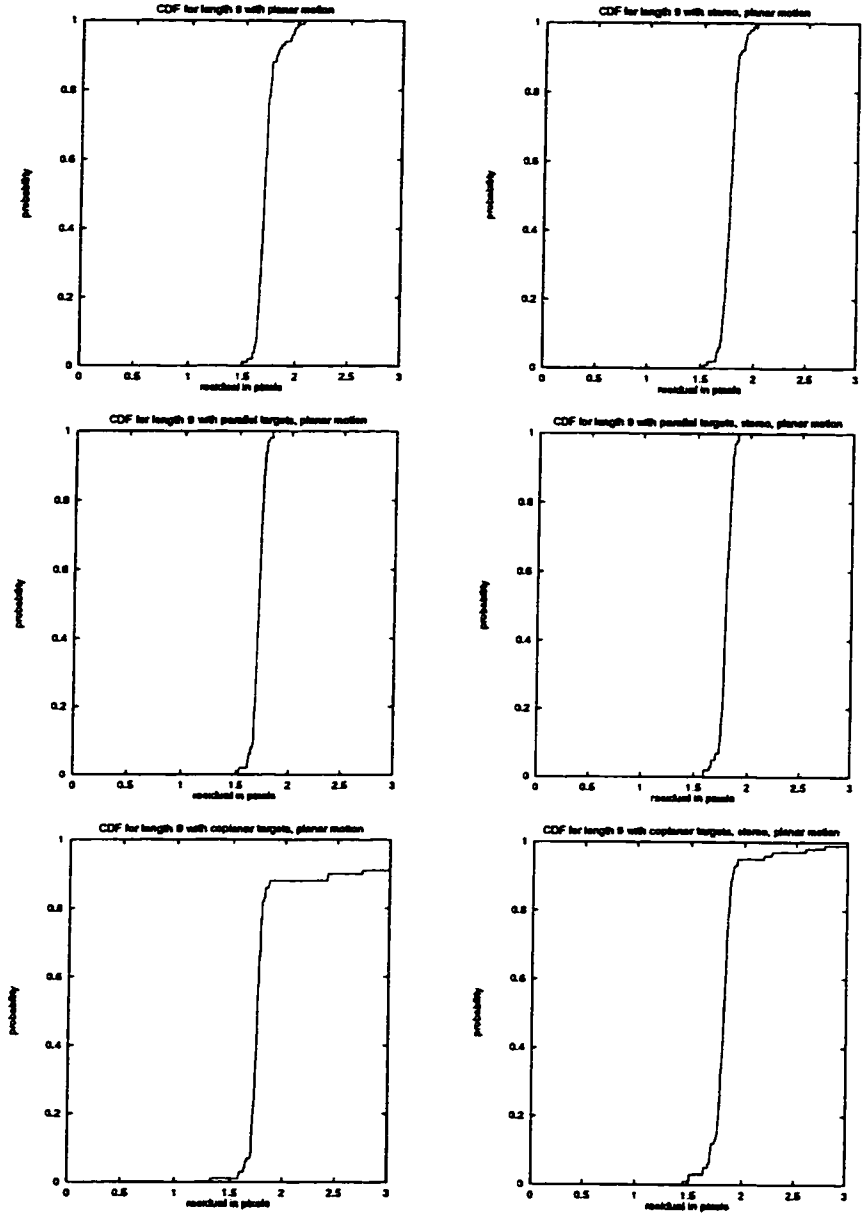


Figure A.6: CDF's for sequences of length nine and planar motion.

Bibliography

[Alo90] J. Aloimonos. Perspective approximations. *Image and Vision Computing*, 8:179–192, 1990.

[Ana89] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[Azu97] R.T. Azuma. A survey of augmented reality. *Presence*, 6:355–385, 1997.

[BAHH92] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proc. ECCV*, pages 237–252, 1992.

[BFB92] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 1992.

[BFBB93] J. Barron, D. Fleet, S. Beauchemin, and T. Burkitt. Performance of optical flow techniques. Technical Report 299, Department of Computer Science, University of Western Ontario, 1993.

[Bla97] S. Blake. 6 dof calibration of a camera with respect to the wrist of a 5-axis machine tool. In *Proc. International Conference on Computer Analysis of Images and Patterns*, pages 191–198, 1997.

[Bur81] P. Burt. Fast filter transforms for image processing. *Computer Graphics and Image Processing*, 16:20–51, 1981.

[Cox89] I. Cox. Blanche: Position estimation for an autonomous robot vehicle. In *Proc. IEEE Int. Workshop on Intelligent Robots and Systems*, pages 432–439, 1989.

[DA89] U. Dhond and J. Aggarwal. Structure from stereo – a review. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(16), 1989.

[DD95] D. DeMenthon and L. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15:123–141, 1995.

[Dru87] M. Drumheller. Mobile robot localization using sonar. *IEEE Pattern Analysis and Machine Intelligence*, 9(2):325–332, 1987.

[EN88] W. Enkelmann and H. Nagel. Investigation of multigrid algorithms for estimation of optical flow fields in image sequences. *Computer Vision, Graphics, and Image Processing*, 43:150–177, 1988.

[Fau93] O. Faugeras. *Three Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, 1993.

[FB81] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24:381–395, 1981.

[FM95] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In *Proc. International Conference on Computer Vision*, pages 951–956, 1995.

[FMS93] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(7):52–62, 1993.

[FS97] H. Farid and E. Simoncelli. Optimally rotation-equivariant directional derivative kernels. In *Proc. International Conference on Computer Analysis of Images and Patterns*, pages 207–214, 1997.

[Gan84] S. Ganapathy. Decomposition of transformation matrices for robot vision. *Pattern Recognition Letters*, 2:401–412, 1984.

[Har93] R. Hartley. Euclidean reconstruction from uncalibrated views. In *Proc. Second Europe-US Workshop on Invariance*, pages 187–202, 1993.

[Har94] R. Hartley. Projective reconstruction from line correspondences. In *Proc. IEEE CVPR*, 1994.

[HCLL89] R. Horaud, B. Conio, O. Leboulleux, and B. Lacolle. An analytic solution for the perspective 4-point problem. *Computer Vision, Graphics, and Image Processing*, 47:33–44, 1989.

[HDLC97] R. Horaud, F. Dornaika, B. Lamiroy, and S. Christy. Object pose: The link between weak perspective, paraperspective, and full perspective. *International Journal of Computer Vision*, 22:173–189, 1997.

[HGC92] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *Proc. IEEE CVPR*, 1992.

[HK94] E. Healey and R. Kondepudy. Radiometric ccd camera calibration and noise estimation. *IEEE Pattern Analysis and Machine Intelligence*, 16(3):267–276, 1994.

[HN91] R. Holt and A. Netravalli. Camera calibration problem: Some new results. *CGVIP - Image Understanding*, 54:368–383, 1991.

[HN94] T. Huang and A. Netraval. Motion and structure from feature correspondences: A review. *Proceedings of the IEEE*, 82:251–268, 1994.

[Hor86] B. Horn. *Robot Vision.* MIT Press, 1986.

[HYH85] Y. Hung, P. Yeh, and D. Harwood. Passive ranging to known planar point sets. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 80–85, 1985.

[KH94] R. Kumar and A. R. Hanson. Robust methods for estimaging pose and a sensitivity analysis. *Computer Vision and Image Understanding*, 60:313–342, 1994.

[LF96] Q. Luong and O. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, pages 43–76, 1996.

[LHP80] H. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. In *Proc. Royal Society of London B*, volume 208, pages 385–397, 1980.

[LKW92] M. Leuttgen, W. Karl, and A. Willsky. Efficient multiscale regularization with applications to the computation of optical flow. *IEEE Transactions on Image Processing*, 1992.

[Low87] D. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

[LT88] R. Lenz and R. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3-d machine vision metrology. *IEEE Pattern Analysis and Machine Intelligence*, 10(5):713–720, 1988.

[LV94] Q. Luong and T. Vieville. Canonic representation for the geometry of multiple projective views. In *Proc. ECCV*, 1994.

[Mel95] J. P. Mellor. Enhanced reality visualization in a surgical environment. Technical Report 1544, Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1995.

[MF92] S. Maybank and O. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, pages 123–151, 1992.

[MSB97] J. Mendelsohn, E. Simoncelli, and R. Bajcsy. Discrete-time rigid motion constrained optical flow assuming planar structure. Technical Report 410, Department of Computer Science, University of Pennsylvania's GRASP Laboratory, 1997.

[ODD93] D. Oberkampf, D. DeMenthon, and L. Davis. Iterative pose estimation using coplanar points. In *Proc. IEEE CVPR*, pages 626–627, 1993.

[Oli98] J. Oliensis. A critique of structure from motion algorithms. In *Proc. Int. Conf. on Computer Vision*, 1998.

[PHYT95] T. Phong, R. Horaud, A. Yassine, and P. Tao. Object pose from 2-d to 3-d point and line correspondences. *International Journal of Computer Vision*, 15:225–243, 1995.

[SAH91] E. Simoncelli, E. Adelson, and D. Heeger. Probability distributions of optical flow. In *Proc. IEEE CVPR*, 1991.

[Sha95] A. Shashua. Algebraic functions for recognition. *IEEE Pattern Analysis and Machine Intelligence*, pages 779–789, 1995.

[Sig85] K. Sigihara. Location of a robot using sparse visual information. *4th International Symposium on Robotics Research*, pages 333–340, 1985.

[SN94] A. Shashua and N. Navab. Relative affine structure: Theory and application to 3d reconstruction from perspective views. In *Proc. IEEE CVPR*, 1994.

[Sor82] D. Sorensen. Newton's method with a model trust region modification. *SIAM Journal of Numer. Analy.*, 19:409–426, 1982.

[Tsa86] R. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proc. IEEE CVPR*, pages 364–374, 1986.

[Tsa89] R. Tsai. Synopsis of recent progress on camera calibration for 3d machine vision. In O. Khatib et al., editor, *The Robotics Review*, pages 147–159. The MIT Press, Cambridge, MA, 1989.

[VF96] T. Vieville and O. Faugeras. The first order expansion of motion equations in the uncalibrated case. *Computer Vision and Image Understanding*, pages 128–146, 1996.

[VZR95] T. Vieville, C. Zeller, and L. Robert. Using collineations to compute motion and structure in an uncalibrated image sequence. *International Journal of Computer Vision*, 1995.

[WS94] R. Wilson and S. Shafer. What is the center of the image? *Journal of the Optical Society of America A*, 11(11):2946–2955, 1994.

[Yua89] J. Yuan. A general photogrammetric method for determining object position and orientation. *IEEE Transactions on Robotics and Automation*, 5:129–142, 1989.

[Zha98] Z. Zhang. Understanding the relationship between the optimization criteria in two-view motion analysis. In *Proc. International Conference on Computer Vision*, 1998.