# LANDING AN UNMANNED AIR VEHICLE: VISION BASED MOTION ESTIMATION AND NONLINEAR CONTROL

by

Omid Shakernia

Memorandum No. UCB/ERL M99/64

15 December 1999

COVER

# LANDING AN UNMANNED AIR VEHICLE: VISION BASED MOTION ESTIMATION AND NONLINEAR CONTROL

by

Omid Shakernia

Memorandum No. UCB/ERL M99/64

15 December 1999

**ELECTRONICS RESEARCH LABORATORY**

College of Engineering
University of California, Berkeley
94720

## Abstract

Landing an Unmanned Air Vehicle:
Vision Based Motion Estimation and Nonlinear Control

by

Omid Shakernia

Master of Science in Electrical Engineering and Computer Sciences

University of California at Berkeley

Professor Shankar Sastry, Chair

Unmanned air vehicles (UAVs) are being used more and more in a number of civilian and military applications, for example remote monitoring of traffic, search and rescue operations, and surveillance. This has generated considerable interest in the control community, mainly due to the fact that the design of UAVs brings to light research questions falling in some of the most exciting new directions for control. One of these directions is the use of computer vision as a sensor in the feedback control loop. In this report, we use computer vision as a feedback sensor to control the landing a UAV onto a landing pad. The vision problem we address here is a special case of the classic *ego-motion estimation problem* since all feature points lie on a planar surface (the landing pad). We study together the *discrete* and *differential* versions of the ego-motion estimation problem in order to obtain both position and velocity of the UAV relative to the landing pad. After briefly reviewing existing the algorithms for the discrete case, we present, in a unified geometric framework, a new estimation scheme for solving the differential case. We further show how the obtained algorithms enable the vision sensor to be placed in the feedback loop as a state observer for landing control. These algorithms are linear, numerically robust, computationally inexpensive and hence suitable for real-time implementation. We present a thorough performance evaluation of the motion estimation algorithms under varying levels of image measurement noise, altitudes of the camera above the landing pad, and different camera motions relative to the landing pad. We also design a landing controller for a full dynamic model of the UAV. Using geometric nonlinear control theory, the dynamics of the UAV are decoupled into an *inner system* and *outer system.* The proposed control scheme is based on the *differential flatness* of the outer system. For the overall closed-loop system, conditions are provided under which exponential stability can be guaranteed. In the closed-loop system, the controller is tightly coupled with the vision based state estimation and the only auxiliary sensor are accelerometers for measuring acceleration of the UAV. Finally, we show through simulation results that the designed vision-in-the-loop controller generates stable landing maneuvers even for large levels of image measurement noise.

*To my family.*

# Contents

**List of Figures** vi

**1 Introduction** 1

**2 Camera Motion and Projection Model** 4

**3 Vision Based Motion Estimation** 6

3.1 Discrete Case . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

3.2 Differential Case . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

3.2.1 Estimating Matrix $B$ . . . . . . . . . . . . . . . . . . . . . . 11

3.2.2 Decomposing Matrix $B$ . . . . . . . . . . . . . . . . . . . . . 13

3.3 Implementation Issues . . . . . . . . . . . . . . . . . . . . . . . . 15

**4 Simulation of Motion Estimation Algorithms** 16

4.1 Depth Sensitivity . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

4.2 Noise Sensitivity . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

4.3 Motion Sensitivity . . . . . . . . . . . . . . . . . . . . . . . . . . 18

**5 Nonlinear Control for a UAV Dynamic Model** 21

5.1 System Dynamics . . . . . . . . . . 22
5.2 Inner and Outer System Partitioning . . . . . . . . . . 23
5.3 Control Design . . . . . . . . . . 25
5.4 Stability Analysis . . . . . . . . . . 27

**6 Vision in the Control Loop** **31**
6.1 Disambiguation of Motion Estimates . . . . . . . . . . 31
6.2 Simulation Results for the Closed-Loop System . . . . . . . . . . 33

**7 Conclusions** **36**

**Bibliography** **39**

**A Proof of Proposition 3.6** **42**

**B UAV System Parameters** **44**

# List of Figures

1.1 A US Navy photo showing a helicopter landing on a ship deck. . . . . . . . . 2

3.1 Geometry of camera frame relative to landing pad. . . . . . . . . . . . . . . . 7

4.1 Depth sensitivity. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

4.2 Noise Sensitivity. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

4.3 Discrete Case: sensitivity to translation-rotation axes. . . . . . . . . . . . . . 19

4.4 Differential Case: sensitivity to translation-rotation axes. . . . . . . . . . . . 20

5.1 Block diagram of UAV dynamics. . . . . . . . . . . . . . . . . . . . . . . . . 22

5.2 Partitioned inner and outer systems. . . . . . . . . . . . . . . . . . . . . . . 25

5.3 Block diagram of control scheme. . . . . . . . . . . . . . . . . . . . . . . . . 26

6.1 Block diagram of vision in control loop. . . . . . . . . . . . . . . . . . . . . 32

6.2 Closed-loop system landing simulation with 0 pixel noise. . . . . . . . . . . . 34

6.3 Closed-loop system landing simulation with 1 pixel noise. . . . . . . . . . . . 34

6.4 Closed-loop system landing simulation with 2 pixels noise. . . . . . . . . . . 35

6.5 Closed-loop system landing simulation with 4 pixels noise. . . . . . . . . . . 35

7.1 Showing screen shots of a 3D visualization of a UAV landing on a ship. Figure courtesy of Cory Sharp. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37

7.2 Showing a member of UC Berkeley UAV fleet. Figure courtesy of Hyunchul Shim. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38

7.3 Showing the pitching landing deck. Figure courtesy of Tullio Celano III. . . . 38

# Acknowledgements

First and foremost, I would like to thank my research adviser, Professor Shankar Sastry, for being a constant source of inspiration and guidance in my research. I am also grateful to Professor Jitendra Malik and Dr. Jana Kosecka, whose courses in computer vision introduced me to this exciting field of research. I am sincerely grateful to Yi Ma, John Koo and Dr. Joao Hespanha, who are research partners in this work. I'd also like to thank the members of the Berkeley Aerobot team: David Shim, Cory Sharp, Rene Vidal, Tullio Celano, Bruno Sinopoli, Jin Kim, Shahid Rashid, William Morrison, Santosh Phillip, Cedric Ma, and Peter Ray for their teamwork, energy and motivation. I'd also like to thank my officemates Joseph Yan, Cenk Cavusoglu, and Jianghai Hu, for providing a stimulating work environment. Finally, I would like to thank my family for their endless love and support. This work is dedicated to them.

# Chapter 1

# Introduction

Unmanned air vehicles (UAVs) are being used more and more in a number of civilian and military applications, for example remote monitoring of traffic, search and rescue operations, and surveillance. This has generated considerable interest in the control community, mainly due to the fact that the design of UAVs brings to light research questions falling in some of the most exciting new directions for control. One of these directions is the use of computer vision as a sensor in the feedback control loop. The task of autonomous aircraft landing is particularly well suited to vision based control, especially in cases where the landing pad is in an unknown location and is moving, such as the deck of a ship. Figure 1.1 shows a US Navy helicopter landing onto the deck of a ship.

Typically, a vision system on board a UAV augments a sensor suite including a Global Positioning System (GPS) which provides position information relative to the inertial frame, and Inertial Navigation Sensors (INS) which provide acceleration information [27]. As a cheap, passive and information-abundant sensor, computer vision is gaining more and more importance in the sensor suite of mobile robots. There has been a growing interest in control design around a vision sensor. In [19], stereo vision systems are proposed to augment a multi-sensor suite including laser range-finders in the landing maneuver of a UAV. In [28], the use of projections of parallel lines is proposed for the purpose of estimating the location and orientation of the helicopter landing pad. Using this approach, the vision sensor provides position and orientation estimates of the camera relative to the landing pad, but can not

Figure 1.1: A US Navy photo showing a helicopter landing on a ship deck.

estimate the camera velocity, which is important for controlling a UAV.

In this report, we present computer vision algorithm based on [20, 21] to estimate UAV motion (position and orientation, linear and angular velocity) relative to a landing pad using a calibrated monocular camera. The given algorithms are linear, computationally inexpensive, numerically robust, and amenable to real-time implementation. We also present a thorough performance evaluation of the vision based motion estimation under varying levels of image measurement noise, altitudes, and camera motions relative to the landing pad.

The use of computer vision in the control of UAVs is more challenging than in the classical "visual servoing" approach [2] because UAVs are under-actuated nonlinear dynamical systems. In order for a guaranteed performance such as stability for the overall closed-loop system, a thorough characterization of the UAV dynamics are absolutely necessary. We present a full dynamic model of the UAV. Based on geometric control theory, we decompose the dynamics into two subsystems: inner and outer systems. A nonlinear controller is proposed based on differential flatness of the outer system. In addition to the work in [24], we also give a detailed stability analysis of the closed-loop system, and clear conditions are derived for system stability. The proposed controller is tightly coupled with the vision based state estimation and the only auxiliary sensor needed to implement the controller is an INS for measurement of acceleration. The INS is used since second order derivatives of image features are highly sensitive to noise. Finally, we show through simulation that the designed vision-in-the-loop controller is stable even for large levels of image measurement noise.

## Report Outline

In chapter 2, we introduce the notation for the camera motion and imaging models. In chapter 3 we formulate the problem of motion estimation from image measurements of a planar scene. We present a new geometrical scheme for recovering of the camera linear and angular velocities from the velocities of feature image points. In chapter 4 we provide simulation results of the planar ego-motion estimation algorithms and evaluate their performance under the presence of noise, and different types of motion relative to the plane. In chapter 5 we give the dynamic model of the UAV and the design of a controller based on differential flatness. Conditions for closed-loop stability are also studied in detail. In chapter 6 we describe how the obtained vision algorithms can be placed in the feedback loop as a state estimator for the controller, and provide simulation results of the vision based landing maneuver. We end the report in chapter 7 with concluding remarks and directions for future research.

# Chapter 2

# Camera Motion and Projection Model

We begin by introducing the notation for the motion of the UAV and the imaging model of the on-board camera. The notation is consistent with that given in [14, 16]. We identify a point with its coordinates in the inertial frame, and use the terms "point" and "coordinates" interchangeably when referring to points in the inertial frame. We will adhere to the following convention: We denote the coordinates of a point in the inertial frame with a tilde, for example $\tilde{q} \in \mathbb{R}^3$, and denote the coordinates of the point in the camera frame using the same letter, but without a tilde, for example $q \in \mathbb{R}^3$. We assume a monocular camera is fixed to the UAV and the optical axis of the camera coincides with the vertical axis of the UAV body frame.

Given a vector $\omega = (\omega_1, \omega_2, \omega_3)^T \in \mathbb{R}^3$, we define $\widehat{\omega} \in so(3)$, the space of skew-symmetric matrices in $\mathbb{R}^{3\times3}$, by:

$$\widehat{\omega} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}. \tag{2.1}$$

It follows from the definition of cross-product of vectors that for any two vectors $p, q \in \mathbb{R}^3$, we have $p \times q = \widehat{p}q$.

We assume that the motion of the UAV is described by a smooth curve on the *special Euclidean group* $SE(3) = \{(p, R) \mid p \in \mathbb{R}^3, R \in SO(3)\}$, where $SO(3)$ is the *special orthog-*

*onal group* of rotation matrices in $\mathbb{R}^{3\times 3}$. Let $(p(t), R(t)) \in SE(3)$ denote the position and orientation of the camera with respect to the inertial frame at time $t$. Then the coordinates of a fixed point in the inertial frame, and its coordinates in the camera frame at time $t$ are related by:

$$\tilde{q} \quad = \quad R(t)q(t) + p(t). \tag{2.2}$$

Since $\tilde{q}$ is fixed, differentiating equation (2.2) leads to $\dot{q} = -R^T\dot{R}q - R^T\dot{p}$. Since $R^TR = I$, we have $R^T\dot{R} = -\dot{R}^TR$, and hence $R^T\dot{R} \in so(3)$. We define $\omega \in \mathbb{R}^3$ and $v \in \mathbb{R}^3$ by:

$$\widehat{\omega} = R^T\dot{R}, \qquad v = R^T\dot{p}. \tag{2.3}$$

With this notation we have:

$$\dot{q} = -\widehat{\omega}q - v. \tag{2.4}$$

The meaning of these variables is that $\omega$ and $v$ are the angular and linear velocities of the camera relative to the inertial frame, given in the instantaneous camera frame.

The imaging of the camera is given by the *perspective projection* of points in the 3D world onto the image plane. We assume a calibrated camera, and without loss of generality we take the image plane to be at a unit distance from optical center of the camera. The perspective projection of the camera is then given by:

$$\begin{aligned} \pi : \mathbb{R}^3 \setminus \{0\} &\to \mathbb{RP}^2 \\ (q_x, q_y, q_z)^T &\mapsto \left(\frac{q_x}{q_z}, \frac{q_y}{q_z}, 1\right)^T. \end{aligned} \tag{2.5}$$

If $\mathbf{x}$ is the image of a point $q$, *i.e.* $\mathbf{x} = \pi(q)$, then we can write:

$$\lambda \mathbf{x} = q \tag{2.6}$$

where $\lambda = q_z \in \mathbb{R}$ encodes the depth of $q$ from the optical center of the camera. Denoting the optical axis by $e_3 = (0, 0, 1)^T$, we have $\lambda = e_3^Tq$. Rewriting equation (2.6), we get the following identity:

$$(I - \mathbf{x}e_3^T)q = 0 \tag{2.7}$$

which will be useful in the later development.

# Chapter 3

# Vision Based Motion Estimation

In this chapter, we give a formulation of the so-called *ego-motion estimation problem* [12, 15, 23, 26]. The goal is to recover the motion of the camera using image measurements of fixed points in the scene. The ego-motion estimation problem can be subdivided into the *discrete* and *differential* cases. In the *discrete* case, one uses the correspondences between image points in a pair of images to compute the rigid camera motion between the frames. A solution to the discrete case can be obtained by the well-known *8-point algorithm* [12]. In the 8-point algorithm, one uses image correspondences of at least eight fixed points in general configuration in the scene and minimizes the *epipolar constraint* to recover the so-called *essential matrix* that contains all the motion parameters. Then singular value decomposition is used to uniquely decompose the essential matrix and find the translation and rotation between the camera frames. The *differential ego-motion estimation problem* is to recover the translational and rotational velocity of the camera given the velocities of image points in a sequence of images. Recently, Ma *et al* [14] derived a counterpart for the 8-point algorithm for the differential case. In a similar spirit to the discrete case, using at least eight image point velocities, one minimizes a *differential epipolar constraint* and recovers a *differential essential matrix*, that contains all the motion parameters. Then eigenvalue-decomposition is used to uniquely decompose the differential essential matrix to find the translational and rotational velocity of the camera.

The ego-motion estimation problem for the purpose of landing a UAV is a special case

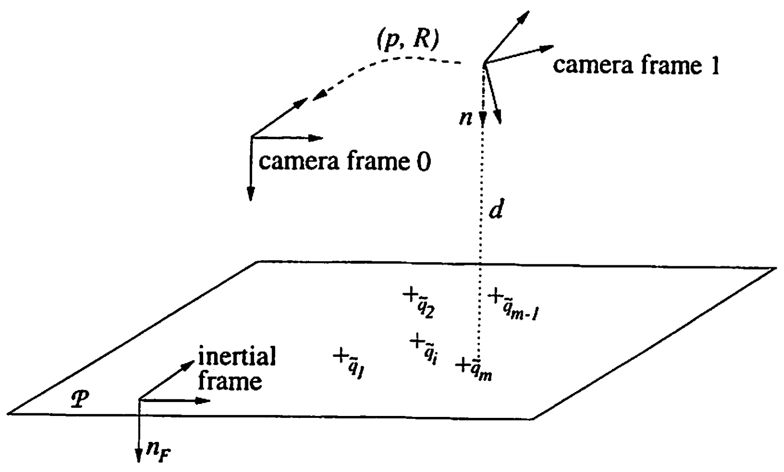


Figure 3.1: Geometry of camera frame relative to landing pad.

of the general one: All the image points correspond to coplanar points on the landing pad. It is well known that the case where all features points in the scene are coplanar is a degenerate case that makes the 8-point algorithm ill-conditioned, giving poor estimation results [4]. Hence one needs algorithms specific to the planar case. The discrete version of the planar ego-motion problem has been studied extensively [3, 13, 26]. Here we only formulate the problem and briefly revisit well-known results that can be found in [26]. Our contribution is to the differential version of the problem. The differential version is important when the task is to control a dynamic mobile robot such as a UAV, since velocity estimates are necessary for the computation of control inputs. The differential planar ego-motion estimation problem has been studied by many researchers [8, 13, 22, 25], each using a different approach. We here propose a new and unified geometric framework which provides motion estimation algorithms for the planar case in the same spirit of the general purpose 8-point algorithm: First minimize a planar discrete (differential) epipolar constraint based on image correspondences (velocities) to obtain a planar discrete (differential) essential matrix, then use SVD (eigenvalue-decomposition) of the essential matrix to recover the unknown motion parameters.

## 3.1 Discrete Case

Suppose we have a set of $m$ fixed coplanar points $\{\tilde{q}_i\}_{i=1}^m \subset \mathcal{P}$, where $\mathcal{P}$ denotes the landing plane. Without loss of generality, we take the origin of the inertial frame to be in $\mathcal{P}$. Figure 3.1 depicts the geometry of the camera frame relative to the landing pad. We will assume throughout the report that the optical center of the camera never passes through the plane. We have the following proposition, which gives a constraint on the coordinates of the coplanar points.

**Proposition 3.1.** *Suppose the camera undergoes a rigid motion* $(p, R) \in SE(3)$. *Then the coordinates* $\{q_i^0\}_{i=1}^m, \{q_i^1\}_{i=1}^m$ *of the fixed coplanar points* $\{\tilde{q}_i\}_{i=1}^m \subset \mathcal{P}$ *in the two camera frames are related by:*

$$q_i^0 = \left(R + \frac{1}{d}pn^T\right) q_i^1, \qquad i = 1, \ldots, m \tag{3.1}$$

*where* $d$ *is the perpendicular distance of camera frame* 1 *to the plane* $\mathcal{P}$ *and* $n \in S^2$ *is the surface normal of* $\mathcal{P}$ *relative to camera frame* 1.

*Proof.* Let $(p_0, R_0), (p_1, R_1) \in SE(3)$ be the configurations of camera frames 0 and 1, respectively. Without loss of generality, we take $R_0 = I$, and hence the rigid motion between the camera frames is $(p, R) = (p_1 - p_0, R)$. For each $i = 1, \ldots, m$ we have:

$$q_i^0 = Rq_i^1 + p \tag{3.2}$$

where $q_i^0, q_i^1$ are the coordinates of $\tilde{q}_i$ in camera frames 0 and 1 respectively. Let $n_F \in S^2$ be the unit normal vector of the plane $\mathcal{P}$ in terms of the inertial frame. Then the surface normal in the coordinate frame of camera 1 is given by $n = R^T n_F$. If $d > 0$ denotes the distance from the plane $\mathcal{P}$ to the optical center of camera frame 1, then we have:

$$\frac{1}{d}n^T q_i^1 = 1, \qquad i = 1, \ldots, m. \tag{3.3}$$

Substitution equation (3.3) into equation (3.2) gives the result. $\square$

We call the matrix:

$$A = \left(R + \frac{1}{d}pn^T\right) \in \mathbb{R}^{3\times 3} \tag{3.4}$$

the *planar essential matrix*, since it contains all the the *motion parameters* $\{p, R\}$ and the *structure parameters* $\{n, d\}$ that we need to recover. Notice that due to the inherent scale ambiguity in the term $\frac{1}{d}p$ in equation (3.4), the vision sensor can in general only recover the ratio of the camera translation scaled by the inverse distance to the plane. In section 6.1 we show how to resolve this ambiguity when the vision sensor is used for landing.

**Lemma 3.2.** *The matrix $A = R + \frac{1}{d}pn^T$ satisfies the constraint:*

$$(I - \mathbf{x}_i^0 e_3^T)A\mathbf{x}_i^1 = 0, \qquad i = 1, \ldots, m \tag{3.5}$$

*where $\{\mathbf{x}_i^0\}_{i=1}^m$, $\{\mathbf{x}_i^1\}_{i=1}^m$, are the images of $\{q_i^0\}_{i=1}^m, \{q_i^1\}_{i=1}^m$ respectively.*

*Proof.* Simply apply equation (2.7) to equation (3.1) □

Equation (3.5) is the *planar discrete epipolar constraint.* Since the constraint given by Lemma 3.2 is *linear* in $A$, by stacking the entries of $A$ as $a = (a_{11}, a_{12}, a_{13}, a_{21}, \ldots, a_{33})^T \in \mathbb{R}^9$, we may re-write equation (3.5) as $\mathbf{f}_i a = 0$, where $\mathbf{f}_i \in \mathbb{R}^{3\times 9}$ is a function of $\mathbf{x}_i^0, \mathbf{x}_i^1$. Since the third row in equation (3.5) is all zeroes, the third row of $\mathbf{f}_i$ contains all zeroes, so we simply drop it and take $\mathbf{f}_i \in \mathbb{R}^{2\times 9}$. With this notation, given $m$ image points correspondences, by defining $\mathbf{F} = (\mathbf{f}_1^T, \ldots, \mathbf{f}_m^T)^T \in \mathbb{R}^{2m\times 9}$ we can combine the equations (3.5) and rewrite them as:

$$\mathbf{F}a = 0. \tag{3.6}$$

In order to solve uniquely (up to scale) for $a$, we must have rank($\mathbf{F}$) = 8. Each pair of image point correspondences gives two constraints, hence we would expect that at least four point correspondences would be necessary for the estimation of $A$. We say a set of coplanar points are in *general configuration* if there is a set of four points such that no three are collinear.

**Proposition 3.3 (Weng [26]).** rank($\mathbf{F}$) = 8 *if and only if the points $\{\tilde{q}_i\}_{i=1}^m$ are in general configuration in the plane.*

Proposition 3.3 says that if there are at least four point correspondences of which no three are collinear, then we may apply standard linear least squares estimation to recover $A$ up its scale. That is, we can recover $A_L = \xi A$ for some unknown $\xi \in \mathbb{R}$. Due to the nature of least squares estimation, as the number of feature points increases, the estimation of the $A$ matrix, and hence the motion estimates, improves.

It turns out that the middle singular value of the matrix $A = R + \frac{1}{d}pn^T$ is identically equal to 1 [3, 26]. Then, if $(\sigma_1, \sigma_2, \sigma_3)$ are the singular values of $A_L$, we set $A = \frac{1}{\sigma_2}A_L$, which determines $A$ up to a sign. To get the correct sign, we use $\lambda_i^0 \mathbf{x}_i^0 = \lambda_i^1 A \mathbf{x}_i^1$ and the fact that $\lambda_i^0, \lambda_i^1 > 0$ to impose the constraint $(\mathbf{x}_i^0)^T A \mathbf{x}_i^1 > 0$ for $i = 1, \ldots, m$. Thus, we have that if the points $\{\tilde{q}_i\}_{i=1}^m$ are arranged in general configuration then the matrix $A$ can be uniquely estimated from the image measurements. Once we have recovered $A$, we need some more SVD analysis in order to decompose it into its motion and structure parameters. For the details on the decomposition please refer to [26]. In general, for a matrix $A = (R + \frac{1}{d}pn^T)$, there are two physically possible solutions for its decomposition into parameters $\{p, R, n, d\}$. In section 6.1 we give a method of disambiguating the solutions when the task is landing a UAV on a landing pad whose geometry is known *a priori*.

## 3.2 Differential Case

Here, in addition to measuring image points, we measure *image point velocities.*

**Proposition 3.4.** *Suppose the camera undergoes a rigid motion with body linear and angular velocities $v(t)$, $\omega(t)$. Then the coordinates of coplanar points $\{\tilde{q}_i\}_{i=1}^m$ in the camera frame satisfy:*

$$\dot{q}_i(t) = -\left(\widehat{\omega} + \frac{1}{d}vn^T\right) q_i(t), \qquad i = 1, \ldots, m. \tag{3.7}$$

*Proof.* Each of the points $q_i$ satisfies:

$$\dot{q}_i = -\widehat{\omega} q_i - v. \tag{3.8}$$

Let $n(t) = R(t)^T n_F$, be the surface normal to $\mathcal{P}$ in the camera frame at time $t$, where $R(t)$ is the orientation of the camera frame. Then, if $d(t) > 0$ is the distance from the optical center of the camera to the plane $\mathcal{P}$ at time $t$, then:

$$\frac{1}{d(t)} n(t)^T q_i(t) \equiv 1, \qquad i = 1, \ldots, m. \tag{3.9}$$

Substituting equation (3.9) into equation (3.8) gives the result. □

We call the matrix:

$$B = \left(\widehat{\omega} + \frac{1}{d} v n^T\right) \in \mathbb{R}^{3\times 3} \tag{3.10}$$

the *planar differential essential matrix*, since it contains all the differential motion parameters $\{v, \omega\}$ and structure parameters $\{n, d\}$ that we need to recover. As in the discrete case, there is an inherent scale ambiguity in the term $\frac{1}{d}v$ in equation (3.10). Thus the vision sensor can in general only recover the ratio of the camera translational velocity scaled by the inverse distance to the plane. In section 6.1 we show how to resolve this ambiguity when the vision sensor is used for landing.

### 3.2.1 Estimating Matrix $B$

We first give a proposition which will be used to prove the main result of this section: Given image velocities of at least four points in general configuration in the plane, we can *uniquely* estimate the planar differential essential matrix.

**Proposition 3.5.** *The matrix $B = (\widehat{\omega} + \frac{1}{d} v n^T)$ satisfies the constraint:*

$$\dot{\mathbf{x}}_i = -(I - \mathbf{x}_i e_3^T) B \mathbf{x}_i, \qquad i = 1, \ldots, m \tag{3.11}$$

*where $\{\mathbf{x}_i(t), \dot{\mathbf{x}}_i(t)\}_{i=1}^m$ are image points and velocities of fixed points $\{\tilde{q}_i\}_{i=1}^m$ in the plane.*

*Proof.* We will drop the subscript $i$ for ease of notation. Differentiating $\lambda \mathbf{x} = q$ and substituting $\dot{q} = -Bq$ gives $\lambda \dot{\mathbf{x}} + \dot{\lambda} \mathbf{x} = -\lambda B \mathbf{x}$. Differentiating $\lambda = e_3^T q$ gives $\dot{\lambda} = -\lambda e_3^T B \mathbf{x}$. Using these relations and eliminating $\dot{\lambda}$ gives the result. □

Equation (3.11) is the *planar differential epipolar constraint.* Since the constraint is linear in $B$, by stacking the entries of $B$ as $b = (b_{11}, b_{12}, b_{13}, b_{21}, \ldots, b_{33})^T \in \mathbb{R}^9$, we may rewrite (3.11) as $\dot{\mathbf{x}}_i = \mathbf{g}_i^T b$, where $\mathbf{g}_i \in \mathbb{R}^{9\times 3}$ is a function of $\mathbf{x}_i$. However, since the third row of equation (3.11) contains only zeros, each image point velocity only imposes two constraints on the matrix $B$. Given a set of $m$ image point and velocity pairs $\{\mathbf{x}_i, \dot{\mathbf{x}}_i\}_{i=1}^m$ of fixed points in the plane, we may stack each equation $\dot{\mathbf{x}}_i = \mathbf{g}_i^T b$ into a single equation:

$$\dot{\mathbf{X}} = \mathbf{G}b \tag{3.12}$$

where $\dot{\mathbf{X}} = (\dot{\mathbf{x}}_1^T, \ldots, \dot{\mathbf{x}}_m^T)^T \in \mathbb{R}^{3m}$ and $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_m)^T \in \mathbb{R}^{3m\times 9}$.

**Proposition 3.6.** rank($\mathbf{G}$) = 8 *if and only if the points* $\{\tilde{q}_i\}_{i=1}^m$ *are in general configuration in the plane.*

*Proof.* Please see to Appendix A. □

If the points are in general configuration in the plane then using linear least squares techniques equation (3.12) can be used to recover $b$ up to one dimension, since $\mathbf{G}$ has a one dimensional null space. That is, we can recover $B = B_L + \xi B_K$ where $B_L$ corresponds to the minimum norm linear least squares estimate of $B$, $B_K$ corresponds to a vector in ker($\mathbf{G}$) and $\xi \in \mathbb{R}$ is an unknown scale. By inspection of equation (3.11) one can see that $B_K = I$. Then we have:

$$B = B_L + \xi I. \tag{3.13}$$

Thus, in order uniquely estimate $B$, we only need to recover the unknown $\xi$. So far, we have not considered the special structure of the $B$ matrix. Next we give constraints imposed by the structure of $B$ which can be used to recover $\xi$, and thus uniquely estimate $B$.

**Lemma 3.7.** *Suppose* $u, v \in \mathbb{R}^3$, *and* $\|u\|^2 = \|v\|^2 = \alpha$. *If* $u \neq v$, *the matrix* $D = uv^T + vu^T \in \mathbb{R}^{3\times 3}$ *has eigenvalues* $\{\lambda_1, 0, \lambda_3\}$, *where* $\lambda_1 > 0$, *and* $\lambda_3 < 0$. *If* $u = \pm v$, *the matrix* $D$ *has eigenvalues* $\{\pm 2\alpha, 0, 0\}$.

*Proof.* Let $\beta = u^T v$. If $u \neq \pm v$, we have $-\alpha < \beta < \alpha$. We can solve the eigenvalues and eigenvectors of $D$ by inspection:

$$\begin{aligned} D(u+v) &= (\beta+\alpha)(u+v) \\ D(u\times v) &= 0 \\ D(u-v) &= (\beta-\alpha)(u-v). \end{aligned}$$

Clearly $\lambda_1 = (\beta+\alpha) > 0$ and $\lambda_3 = \beta - \alpha < 0$. It is direct to check the conditions on $D$ when $u = \pm v$. □

**Theorem 3.8.** *The matrix $B$ can be uniquely estimated from the image measurements if and only if there are four points of $\{\tilde{q}_i\}_{i=1}^m$ in the plane such that no three are collinear.*

*Proof.* In this proof, we will work with sorted eigenvalues, that is if $\{\lambda_1, \lambda_2, \lambda_3\}$ are eigenvalues of some matrix, then $\lambda_1 \geq \lambda_2 \geq \lambda_3$. If the points are not in general configuration, then by Proposition 3.6, rank(G) $< 8$, and the problem is under-constrained. Now suppose the points are in general configuration. Then by least squares estimation we may recover $B = B_L + \xi I$ for some unknown $\xi \in \mathbb{R}$. By Lemma 3.7, we have that $B + B^T = \frac{1}{d}vn^T + \frac{1}{d}nv^T$ has eigenvalues $\{\lambda_1, \lambda_2, \lambda_3\}$ where $\lambda_1 \geq 0$, $\lambda_2 \equiv 0$, and $\lambda_3 \leq 0$. Compute the eigenvalues of $B_L + B_L^T$ and denote them as $\{\gamma_1, \gamma_2, \gamma_3\}$. Since we have $B = B_L + \xi I$, then $\lambda_i = \gamma_i + 2\xi$, for $i = 1, 2, 3$. Since we must have $\lambda_2 = 0$, we have $\xi = -\frac{1}{2}\gamma_2$, and set $B = B_L - \frac{1}{2}\gamma_2 I$. □

### 3.2.2 Decomposing Matrix $B$

We now address the task of decomposing $B$ into its motion and structure parameters. The following constructive proof gives a new technique for the recovery of motion and structure parameters.

**Theorem 3.9.** *Given a matrix $B \in \mathbb{R}^{3\times 3}$ in the form $B = \widehat{\omega} + \frac{1}{d}vn^T$, one can recover the motion and structure parameters $\{\widehat{\omega}, \frac{v}{d}, n\}$ up to at most 2 physically possible solutions. There is a unique solution if $v = 0$, $v \times n = 0$ or $e_3^T v = 0$, where $e_3 = (0, 0, 1)^T$ is the optical axis.*

*Proof.* Compute the eigenvalue/eigenvector pairs of $B + B^T$ and denote them as $\{\lambda_i, u_i\}$, $i = 1, 2, 3$. If $\lambda_i = 0$ for $i = 1, 2, 3$, then we have $v = 0$ and $\widehat{\omega} = B$. In this case we can not recover the normal of the plane $n$. Otherwise, if $\lambda_1 > 0$, and $\lambda_3 < 0$, then we have $v \times n \neq 0$. Let $\alpha = \|v/d\| > 0$, let $\tilde{v} = v/\sqrt{\alpha}$ and $\tilde{n} = \sqrt{\alpha}n$, and let $\beta = \tilde{v}^T\tilde{n}$. According to Lemma 3.7, the eigenvalue/eigenvector pairs of $B + B^T$ are given by:

$$\begin{cases} \lambda_1 = \beta + \alpha > 0, & u_1 = \frac{1}{\|\tilde{v}+\tilde{n}\|}(\tilde{v} + \tilde{n}) \\ \lambda_3 = \beta - \alpha < 0, & u_3 = \frac{1}{\|\tilde{v}-\tilde{n}\|}(\tilde{v} - \tilde{n}). \end{cases} \tag{3.14}$$

Then $\alpha = \frac{1}{2}(\lambda_1 - \lambda_3)$. It is direct to check that $\|\tilde{v} + \tilde{n}\|^2 = 2\lambda_1$, $\|\tilde{v} - \tilde{n}\|^2 = -2\lambda_3$. Then together with (3.14), we have a solution:

$$\left\{ \begin{array}{rcl} \tilde{v}_1 & = & \frac{1}{2}(\sqrt{2\lambda_1}\, u_1 + \sqrt{-2\lambda_3}\, u_3) \\ \tilde{n}_1 & = & \frac{1}{2}(\sqrt{2\lambda_1}\, u_1 - \sqrt{-2\lambda_3}\, u_3) \\ \widehat{\omega}_1 & = & \frac{1}{2}((B - \tilde{v}_1\tilde{n}_1^T) - (B - \tilde{v}_1\tilde{n}_1^T)^T). \end{array} \right. \tag{3.15}$$

The estimate of $\widehat{\omega}_1$ is computed as above because, in the presence of noise, in general $B - \tilde{v}_1\tilde{n}_1^T$ is not necessarily an element in $so(3)$. We here take the projection of $B - \tilde{v}_1\tilde{n}_1^T$ onto $so(3)$.

However, the eigenvalue-decomposition $\{\lambda_i, u_i\}$ is not unique - there is a sign ambiguity in the eigenvectors $u_1$ and $u_3$. This sign ambiguity leads to a total of 4 possible solutions for $\tilde{v}$ and $\tilde{n}$ computed according to (3.15). It is direct to check that that if $\{\widehat{\omega}, \frac{v}{d}, n\}$ are the true motion and structure parameters, then the 4 possible solutions obtained by (3.15) are:

| | | | |
|---|---|---|---|
| Solution 1 (true) | $v_1 = v/d$ | Solution 3 | $v_3 = -v_1$ |
| | $n_1 = n$ | | $n_3 = -n_1$ |
| | $\widehat{\omega}_1 = \widehat{\omega}$ | | $\widehat{\omega}_3 = \widehat{\omega}_1$ |
| Solution 2 | $v_2 = \|v/d\| n$ | Solution 4 | $v_4 = -v_2$ |
| | $n_2 = \frac{1}{\|v/d\|} v/d$ | | $v_4 = -n_2$ |
| | $\widehat{\omega}_2 = \widehat{\omega} - nv^T/d + vn^T/d$ | | $\widehat{\omega}_3 = \widehat{\omega}_2$ |

In order to reduce the number of physically possible solutions, we impose the so-called "positive depth constraint" - since the camera can only see points that are in front of it, we must have $n^T e_3 > 0$. This constraint eliminates solution 3 as being physically impossible. If $v^T e_3 \neq 0$, one of solutions 2 or 4 will be eliminated, whereas if $v^T e_3 = 0$ both solutions 2 and 4 are eliminated. For the case that $v \times n = 0$, it is easy to see that solutions 1 and 2 are equivalent, and that imposing the positive depth constraint leads to a unique solution. $\square$

The results for the ambiguities of solutions were also reported in [8, 22, 25]. In section 6.1 we give a method of disambiguating the solutions when the task is landing a UAV on a landing pad whose geometry is known *a priori*.

## 3.3 Implementation Issues

For both the discrete and differential algorithms, the most computationally intensive task is the linear least squares estimation of the $A$ and $B$ matrices, which involves the singular value decomposition (SVD) of the matrices $\mathbf{F}, \mathbf{G} \in \mathbb{R}^{2m \times 9}$ where $m$ is the number of tracked feature points. The cost of the SVD of a matrix $M \in \mathbb{R}^{n \times m}$ for $n \leq m$ is $O(n^2 m)$ flops. Then, as the number of tracked feature points $m$ increases, the cost of the vision algorithms grows as $O(m)$.

We have implemented the above algorithms using the MATHLIB C++ library in Matlab, and have found that on a 450 MHz Pentium II running Linux, the vision algorithms can perform motion estimation based on 25 tracked feature points at a rate of over 150 Hz. This rate is far beyond that of most current real-time feature tracking hardware.

# Chapter 4

# Simulation of Motion Estimation Algorithms

Since our goal is to use the estimated motion and structure from the vision as a sensor in a control loop, of utmost consideration is the performance of this sensor in the presence of noise in the measurements of point correspondences and image velocity. Another important criteria to analyze is how the estimation errors depend on different camera motions with respect to the observed plane. To this end, we have implemented both the discrete and differential algorithms and performed various simulations in order to evaluate their performance. In order to assess the performance of the planar algorithms, for all simulations we compare the results with the traditional 8-point algorithm as described in [14].

For all simulations, we generated 50 random points uniformly distributed within the field of view of the camera, FOV = $60°$. The image correspondences and the image velocity measurements were corrupted by additive white Gaussian noise. For evaluating the 8-point algorithm, we randomly scattered the depths of these points uniformly between distance of `zmin` and `zmax` focal lengths, where for all simulations, we set `zmax = 400` and `zmin = 100` unless otherwise noted. For evaluating the planar algorithm, we placed the points on the fronto-parallel plane at a distance of `(zmax+zmin)/2`. Each data point on each plot is the mean result of 50 trials for a given motion, noise level, and distance. We studied the performance of the algorithms as a function of depth variation, noise in the image measurements, and motion about different translation/rotation axes.

## 4.1 Depth Sensitivity

In planar case, our depth variation analysis attempts to see how the errors in the estimates depend on the depth of the plane being viewed. Notice that in the matrices $A = R + \frac{1}{d}pn^T$ and $B = \widehat{\omega} + \frac{1}{d}vn^T$, the translation term is scaled by the inverse distance of the plane. Thus, for a fixed translation and a fixed noise level, as the distance of the plane increases, the "signal" from the translation term decreases while the noise level stays constant. Thus, one would expect that as the signal to noise ratio decreases, the performance of the algorithms also decrease. Also, from the structure of the $A$ and $B$ matrices, we see that the errors in the rotational components should not depend on the depths of the points. This expectation was validated as shown in Figure 4.1.

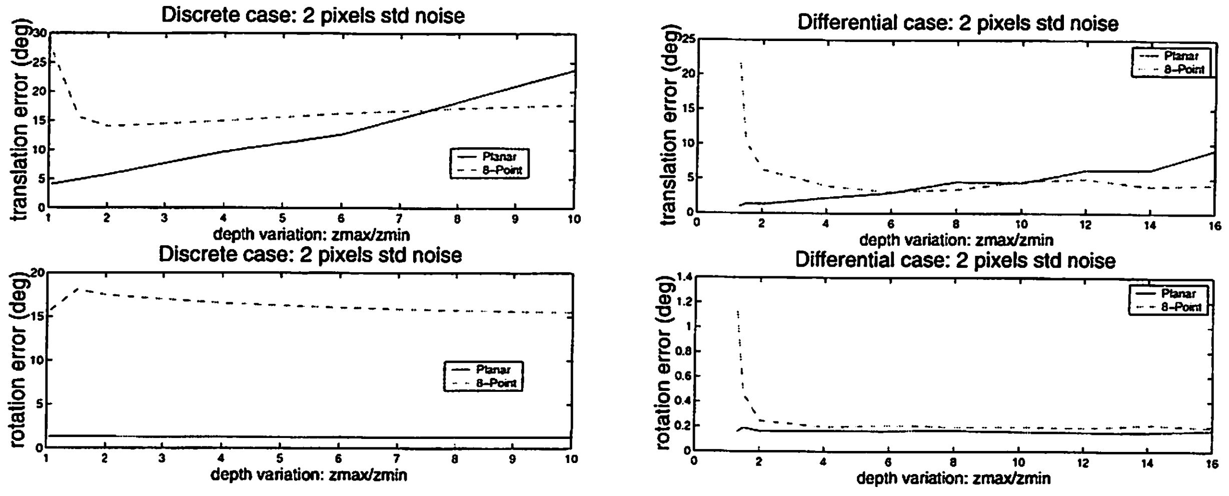


Figure 4.1: Depth sensitivity.

Notice that for very low depth variation, the 8-point algorithm for both discrete and differential case performs poorly. This is a result of singularities that occur in the algorithm when the feature points are coplanar. Also, notice that for the planar case, as expected, the errors increase as the distance of the plane increases. One interesting observation is that for the discrete case, the rotation estimate is always better in the planar case than in the general case.

## 4.2 Noise Sensitivity

In the simulations presented in Figure 4.2, for a given motion we corrupted the correspondences and image velocities with increasing levels additive white Gaussian noise. Notice from the simulation results that for both discrete and differential cases, the planar algorithm performs better that the 8-point algorithm.

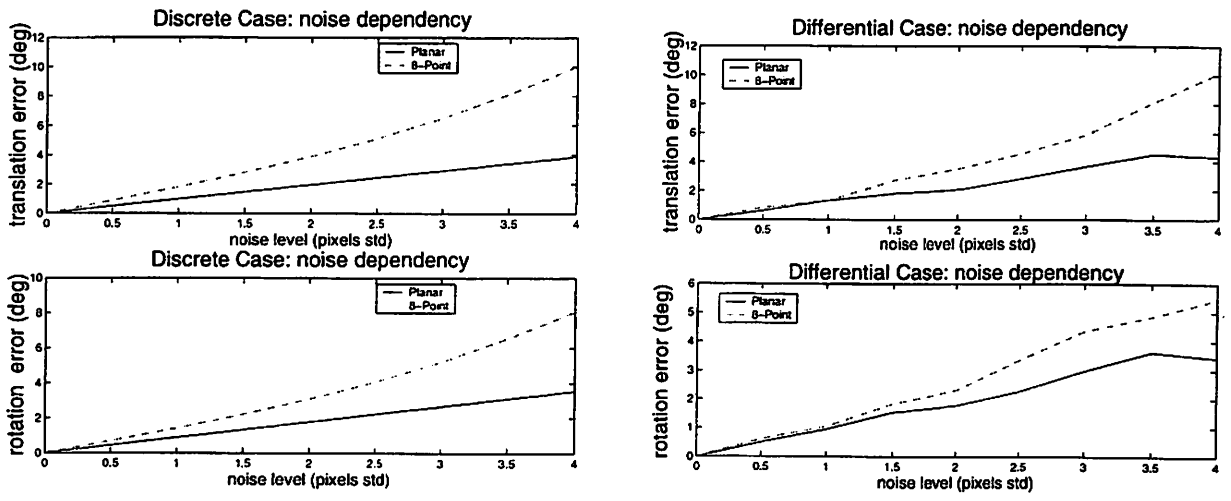


Figure 4.2: Noise Sensitivity.

## 4.3 Motion Sensitivity

Next we study the sensitivity of the algorithm with respect to different motions relative to the plane. We ran the algorithms for a motion about each translation-rotation axis pair for two different noise levels (low and high). In general, the planar algorithms perform better than the 8-point algorithms except when the translation axis is parallel optical axis (and hence the surface normal of the plane). The higher sensitivity in that case can seen as an overall numerical sensitivity to perturbations in the algebraic eigenvalue/eigenvector problem when there are repeated eigenvalues. For example, if a matrix has a pair of repeated eigenvalues then any vector in certain two dimensional subspace can be considered an eigenvector corresponding to the repeated eigenvalue. Because the eigenvectors corresponding to repeated eigenvalues are defined up to subspace, it is intuitive to see that for two different perturbations of the matrix, the corresponding eigenvectors could be quite different. A similar phenomenon occurs in the case of repeated singular vectors. Thus, an algorithm that uses the computation of eigenvectors (singular vectors) is inherently sensitive to noise in the case of repeated eigenvalues (singular values).

The situation of having repeated eigenvalues (singular values) occurs in the planar differential (discrete) algorithm in the case that the translational motion is parallel to the surface normal of the plane. In the 8-point algorithm, the situation of repeated eigenvalues occurs in the case that the translation and rotation axes are parallel [14]. The simulation results for both the discrete (in Figure 4.3) and the differential case (in Figure 4.4) validate our expectation of higher noise sensitivity in the case of repeated singular values and eigenvalues.

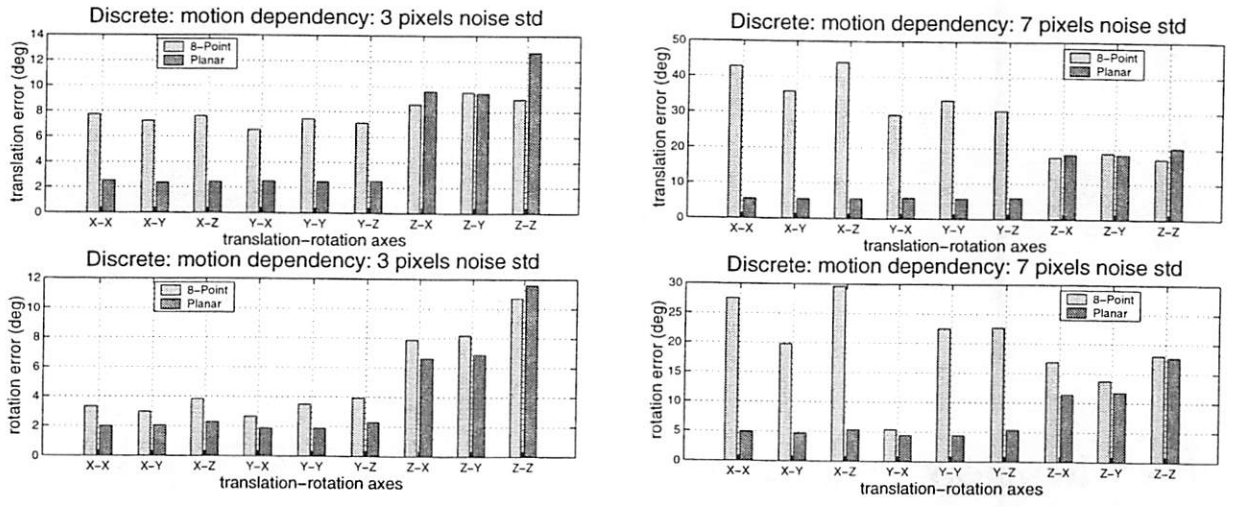


Figure 4.3: Discrete Case: sensitivity to translation-rotation axes.

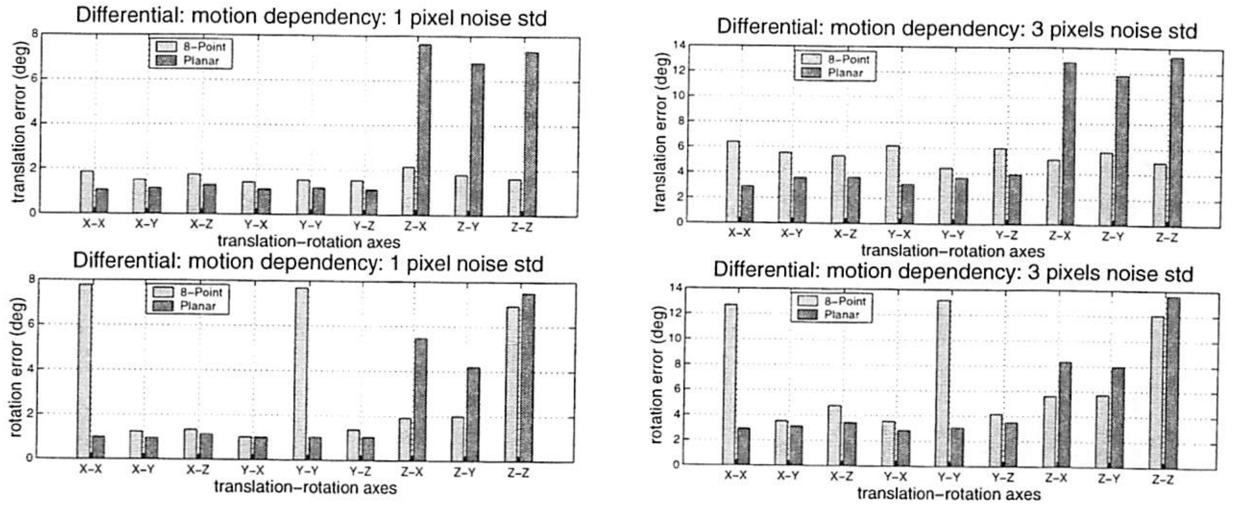


Figure 4.4: Differential Case: sensitivity to translation-rotation axes.

# Chapter 5

# Nonlinear Control for a UAV Dynamic Model

In this chapter, we present the dynamical model of the UAV, a control design based on differential flatness, and a stability analysis of the closed-loop system. The proposed controller is general in the sense that it can be applied towards trajectory tracking. For the purpose of landing, the UAV is asked to track a fixed point at the desired configuration above the landing pad.

We parameterize the orientation $R \in SO(3)$ of the UAV relative to the inertial frame by the $ZYX$ (or "roll, pitch, yaw") Euler angles denoted by $\Theta = (\phi, \theta, \psi)^T$. Thus we have $R = \exp(\widehat{e}_3\psi)\exp(\widehat{e}_2\theta)\exp(\widehat{e}_1\phi)$ with $e_1 = (1,0,0)^T$, $e_2 = (0,1,0)^T$, $e_3 = (0,0,1)^T$. Under this parameterization, there is a mapping $\Psi(\Theta) \in \mathbb{R}^{3\times3}$ given by:

$$\Psi(\Theta) = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{pmatrix} \tag{5.1}$$

which maps the body rotational velocity to Euler angle velocity, that is: $\dot{\Theta} = \Psi\omega$.

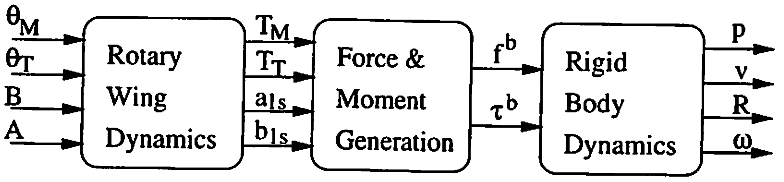


Figure 5.1: Block diagram of UAV dynamics.

## 5.1 System Dynamics

A complete model of a helicopter can be divided into four different subsystems: *actuator dynamics, rotary wing dynamics, force and moment generation processes*, and *rigid body dynamics*. The dynamics of the engine and actuators (which depend on the flexibility of the rotors and fuselage) are quite complex and intractable for analysis. In this report, we consider a helicopter model including only the rigid body dynamics, the force and moment generation process and a simplified rotary wing dynamics. This model is illustrated in Figure 5.1.

We now articulate each of the three subsystems. First, the equations describing the *rigid body dynamics* are given by:

$$\left\{\begin{array}{lcl} \ddot{p} & = & \frac{1}{m} R f^b \\ \dot{\Theta} & = & \Psi \omega \\ \dot{\omega} & = & \mathcal{I}^{-1}(\tau^b - \omega \times \mathcal{I}\omega) \end{array}\right. \tag{5.2}$$

where $m > 0$ is the body mass, $\mathcal{I} \in \mathbb{R}^{3\times 3}$ is the inertial matrix and $f^b, \tau^b \in \mathbb{R}^3$ are the *body force* and *torque* given by:

$$\left\{\begin{array}{lcl} f^b & = & \begin{pmatrix} X_M \\ Y_M + Y_T \\ Z_M \end{pmatrix} + R^T \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix} \\ \tau^b & = & \begin{pmatrix} R_M \\ M_M + M_T \\ N_M \end{pmatrix} + \begin{pmatrix} Y_M h_M + Z_M y_M + Y_T h_T \\ -X_M h_M + Z_M l_M \\ -Y_M l_M - Y_T l_T \end{pmatrix}. \end{array}\right. \tag{5.3}$$

The body forces and torques generated by the main rotor are controlled by $T_M$, $a_{1s}$ and $b_{1s}$, in which $a_{1s}$ and $b_{1s}$ are the longitudinal and lateral tilt of the tip path plane of the main rotor with respect to the shaft, respectively. The tail rotor is considered as a source of pure lateral force $Y_T$ and anti-torque $Q_T$, which are controlled by $T_T$. The forces and torques can be expressed as:

$$\left\{\begin{array}{lcl lcl} X_M & = & -T_M \sin a_{1s} & R_M & \simeq & \frac{\partial R_M}{\partial b_{1s}} b_{1s} - Q_M \sin a_{1s} \\ Y_M & = & T_M \sin b_{1s} & M_M & \simeq & \frac{\partial M_M}{\partial a_{1s}} a_{1s} + Q_M \sin b_{1s} \\ Z_M & = & -T_M \cos a_{1s} \cos b_{1s} & N_M & \simeq & -Q_M \cos a_{1s} \cos b_{1s} \\ Y_T & = & -T_T & M_T & = & -Q_T. \end{array}\right. \tag{5.4}$$

The moments generated by the main and tail rotor can be calculated from the constants $\{l_M, y_M, h_M, h_T, l_T\}$, where $h_i$, $l_i$ and $y_i$ denote the vertical, longitudinal, and lateral distance between the center of gravity and the center of the rotor specified by $i = M$ or $T$. These system parameters are given in Appendix B. In the simulation, we will approximate the rotor torque equations by $Q_i \simeq C_i^Q T_i^{1.5} + D_i^Q$ for $i = M, T$, with details described in [11]. The values of $C_i^Q, D_i^Q$ are also given in Appendix B.

Finally, the *rotary wing dynamics* are in general harder to express explicitly. In an operating region near hovering, the rotary wing dynamics can be approximated by the following equations (for details see [17]):

$$T_M = c_{M1}\theta_M + c_{M3}\theta_M^3, \quad T_T = c_{T1}\theta_T + c_{T3}\theta_T^3, \quad a_{1s} = -B, \quad b_{1s} = A$$

where $\theta_M, \theta_T$ are the main and tail rotor collective pitch, and $B, A$ are the longitudinal and lateral cyclic pitch.

## 5.2 Inner and Outer System Partitioning

A system $\dot{x} = f(x, t, u)$ is called *differentially flat* if there exist output functions, called *flat outputs*, such that all states and inputs can be expressed in terms of the flat outputs and their derivatives [6]. Differential flatness has been applied to approximate models of aircraft [5] and helicopter [10] for trajectory generation. The full helicopter dynamics are not flat in general, however it can be shown that the dynamics can be partitioned into an "inner system" (*e.g.* the attitude dynamics) and an "outer system" (*e.g.* the position dynamics) where the outer system is flat. This scheme has been successfully used for generating a two stage control synthesis for many systems which are not completely flat [24]. Such a scheme which utilizes the flatness of the outer system is roughly illustrated in Figure 5.2. In the figure, $P_O$ is the outer system which is flat, and $P_I$ is the inner system which is not necessarily flat. Given a desired output trajectory, say $y_d^O(\cdot)$, the mapping $F$ in Figure 5.2 utilizes the flatness property of the outer system to generate an desired output trajectory $y_d^I(\cdot)$ for the inner system. The control synthesis for the overall system then reduces to the design of an inner system controller, $C$, which drives the inner system output $y^I(t) \to y_d^I(t)$ (exponentially) as $t \to \infty$. As the inner system output converges, one can show that the outer system output converges to the desired one, $y^O(t) \to y_d^O(t)$ as $t \to \infty$. That is, the overall system asymptotically tracks the desired trajectory.

It has been shown in [10] that the helicopter dynamics are *approximately* differentially flat with the position and heading $\{p, \psi\}$ as the flat outputs. The approximation is based on the assumption that the coupling terms $a_{1s}, b_{1s}, T_T$ are small and can be neglected in the model. So if $a_{1s}, b_{1s}, T_T \approx 0$, the outer system dynamics (5.2) can be rewritten as:

$$\ddot{p} = \frac{1}{m}R(\Theta)\begin{pmatrix} 0 \\ 0 \\ -T_M \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + h \tag{5.5}$$

with

$$h = \frac{1}{m}R(\Theta)\begin{pmatrix} -T_M \sin a_{1s} \\ T_M \sin b_{1s} - T_T \\ -T_M(\cos a_{1s} \cos b_{1s} - 1) \end{pmatrix}.$$

where the inputs are $u^O = y^I = (\Theta, T_M)$, and the outputs are $y^O = (p, \dot{p}, \ddot{p}, \psi)$. One must notice that this approximation introduces a small non-vanishing modeling error $h$ which depends on $\Theta, T_M, a_{1s}, b_{1s}, T_T$. We will soon show its effect on the stability of the closed-loop system.

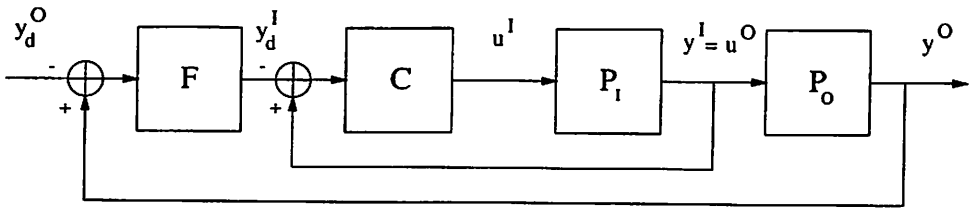


Figure 5.2: Partitioned inner and outer systems.

## 5.3 Control Design

The control design for the overall system is be based on an assumption that there exists a controller $C$ such that $e^I = 0$ is an exponentially stable equilibrium point for the inner error system:

$$\dot{e}^I = f(e^I, e^O, t)|_{e^O=0}, \quad f(0,0,t) = 0$$

where $e^O = y^O - y_d^O$ and $e^I = y^I - y_d^I$ . There have been various design methodologies proposed for the controller of the inner system, *e.g.* [11]. Here, we are only interested in the performance of the overall system assuming such a controller $C$ is already available. As shown in [10], for the approximated outer system (5.5), there exists a smooth mapping from the outer system output to the inner system output:

$$\begin{aligned} \Phi : \ \mathbb{R}^4 &\rightarrow \mathbb{R}^4 \\ (\ddot{p}, \psi) &\mapsto (\Theta, T_M) \end{aligned}$$

which is defined by the equations:

$$\begin{aligned} T_M &= m\sqrt{(\ddot{p}_x)^2 + (\ddot{p}_y)^2 + (\ddot{p}_z - g)^2} \\ \phi &= \sin^{-1}\left(\frac{-\ddot{p}_x \sin\psi + \ddot{p}_y \cos\psi}{T_M/m}\right) \\ \theta &= \text{atan2}\left(\frac{\ddot{p}_x \cos\psi + \ddot{p}_y \sin\psi}{-T_M \cos\phi/m}, \frac{\ddot{p}_z - 1}{-T_M \cos\phi/m}\right) \\ \psi &= \psi \end{aligned}$$

where $\phi, \theta \neq \pm\pi/2$. Suppose that the desired output trajectory of the outer system is $y_d^O = (p_d, \dot{p}_d, \ddot{p}_d, \psi_d)$. To obtain the desired trajectory of the inner system, we define a pseudo-input vector:

$$v_p = \ddot{p}_d + K_v(\dot{p} - \dot{p}_d) + K_p(p - p_d) \tag{5.6}$$

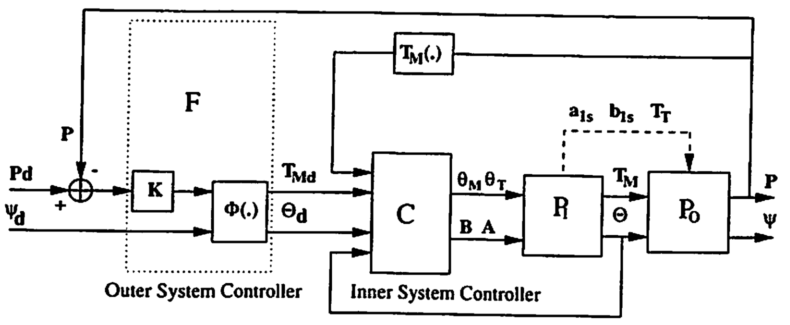


Figure 5.3: Block diagram of control scheme.

where $K_p$, $K_v \in \mathbb{R}^{3\times 3}$ are control parameters. With the above pseudo-input, the desired output of the inner system $y_d^I$ is given by:

$$(\Theta_d, T_{Md}) = \Phi(v_p, \psi_d). \tag{5.7}$$

A more detailed schematic of the controller for this system is illustrated in Figure 5.3. Clearly, if the inner system *exactly* tracks the desired trajectory $(\Theta_d, T_{Md})$, that is, $y_d^I = y^I$ in Figure 5.2, then the behavior of the overall closed-loop system is specified by the outer system only, which, due to chosen the control law (5.6), is *approximately* a linear system with poles assigned by the parameters $K_v, K_p$.

Now if we summarize all conditions so far and rewrite the dynamics of the overall closed-loop system in terms of the tracking errors $e^I$ and $e^O$ of the inner and outer systems respectively, they have the form:

$$\left\{ \begin{array}{lcl} \dot{e}^I & = & f(e^I, e^O, t) \\ \dot{e}^O & = & Ae^O + g(e^I, t) + h(e^I, e^O, t) \end{array} \right. \tag{5.8}$$

where

$$g(e^I, t) = \frac{1}{m} R(\Theta) \begin{pmatrix} 0 \\ 0 \\ -T_M \end{pmatrix} - \frac{1}{m} R(\Theta_d) \begin{pmatrix} 0 \\ 0 \\ -T_{Md} \end{pmatrix}.$$

In the above equations, $f(e^I, e^O, t)$ is in general a function of both $e^I$ and $e^O$ since the input of the inner system is a function of $e^O$. The function $h(e^I, e^O, t)$ from (5.5) is a small non-vanishing approximation error, and $g(e^I, t)$ vanishes when the inner system exactly tracks the desired trajectory, *i.e.*, $g(0,t) = 0$. Since the helicopter model is smooth and many of the parameters are physically bounded, $g(e^I,t)$ is in fact (globally) bounded as $\|g(e^I,t)\| \le L\|e^I\|$ for some constant $L > 0$.[1]

## 5.4 Stability Analysis

We now analyze the performance of the overall closed-loop system. As we have argued before, the function $f$ in (5.8) is in general a function of both $e^I$ and $e^O$. However, in practice, the inner system is usually designed to have a much faster convergence rate than the outer system. To simplify the analysis, for now we assume that the inputs $T_{Md}(\cdot)$ and $\Theta_d(\cdot)$ of the inner system are approximately constant, and thus $f$ is only a function of $e^I$ (the more general case will be presented afterwards).

Recall that given an general system $\dot{x} = f(x,t)$, by the Lyapunov theorem and its converse [18], the system is exponentially stable if and only if there exists a *Lyapunov function* $V(x,t)$ satisfying:

$$\alpha_1\|x\|^2 \le V(x,t) \le \alpha_2\|x\|^2 \tag{5.9}$$
$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x}f(x,t) \le -\alpha_3\|x\|^2 \tag{5.10}$$
$$\left\|\frac{\partial V}{\partial x}\right\| \le \alpha_4\|x\| \tag{5.11}$$

for some positive *Lyapunov constants* $\alpha_1, \alpha_2, \alpha_3, \alpha_4 > 0$. We can apply this theorem to both the nominal outer system $\dot{e}^O = Ae^O$ and the inner system $\dot{e}^I = f(e^I,t)$ and denote the corresponding Lyapunov functions as $V^O$ and $V^I$ and the Lyapunov constants as $\alpha_1, \alpha_2, \alpha_3, \alpha_4 > 0$ and $\beta_1, \beta_2, \beta_3, \beta_4 > 0$ respectively.

**Theorem 5.1.** *Consider the following system:*

$$\begin{cases} \dot{e}^I = f(e^I,t) \\ \dot{e}^O = Ae^O + g(e^I,t) \end{cases} \tag{5.12}$$

*where $g(e^I,t)$ is a perturbation term that satisfies $\|g(e^I,t)\| \le L\|e^I\|$. If both the nominal outer system $\dot{e}^O = Ae^O$ and inner system $\dot{e}^I = f(e^I,t)$ are exponentially stable, then the overall system is exponentially stable for any $L > 0$.*

[1]Such a $L$ can be estimated from the system equation (5.2).

*Proof.* Apply the converse Lyapunov theorem to both the outer and inner systems, and denote the corresponding Lyapunov functions as $V^O, V^I$ and the constants as $\{\alpha_i\}_{i=1}^4, \{\beta_i\}_{i=1}^4$ respectively. We consider the candidate Lyapunov function $V = V^I + \mu V^O$ for the overall system. Then we have:

$$\begin{aligned} \dot{V} = \dot{V}^I + \mu \dot{V}^O &\leq -\beta_3 \|e^I\|^2 - \mu\alpha_3 \|e^O\|^2 + \mu\alpha_4 L \|e^O\| \|e^I\| \\ &= -(\|e^I\|, \|e^O\|) Q (\|e^I\|, \|e^O\|)^T \end{aligned}$$

where the matrix $Q \in \mathbb{R}^{2\times 2}$ is:

$$Q = \begin{pmatrix} \beta_3 & -\frac{1}{2}\mu\alpha_4 L \\ -\frac{1}{2}\mu\alpha_4 L & \mu\alpha_3 \end{pmatrix}.$$

The matrix $Q$ can be positive definite if and only if there exists a small enough $\mu > 0$ such that $\det(Q) > 0$. It is easy to check that it suffices to have $0 < \mu < \frac{4\beta_3\alpha_3}{\alpha_4^2 L^2}$. Such a $\mu$ always exists. Therefore, the overall system is always exponentially stable regardless of $L$. □

This theorem states a very interesting fact for the system (5.12): as long as the inner system and outer system are exponentially stable, the system is extremely robust (in terms of exponential stability) to any (vanishing) perturbation of the outer system which only depends on the tracking error of the inner system.

In the above theorem we assumed that the inner system does not depend on the tracking error $e^O$ of the outer system. For the more general case, we may write:

$$f(e^I, e^O, t) = f(e^I, 0, t) + d(e^I, e^O, t)$$

where $d(e^I, e^O, t) = f(e^I, e^O, t) - f(e^I, 0, t)$. The nominal system $\dot{e}^I = f(e^I, 0, t)$ is exponentially stable as designed and we still denote its Lyapunov function as $V^I$ and Lyapunov constants as $\{\beta_i\}_{i=1}^4$. Then for the overall system, following the spirit of Theorem 5.1, we have the result:

**Theorem 5.2.** *Consider the following perturbed system:*

$$\begin{cases} \dot{e}^I &= f(e^I, e^O, t) &= f(e^I, 0, t) + d(e^I, e^O, t) \\ \dot{e}^O &= Ae^O + g(e^I, t) \end{cases} \tag{5.13}$$

*where $g(e^I, t)$ is a perturbation term that satisfies $\|g(e^I, t)\| \le L_1\|e^I\|$ for some $L_1 > 0$. If, for $d(e^I, e^O, t)$, there exists $L_2 > 0$ such that $\|d(e^I, e^O, t)\| \le L_2\|e^O\|$, then the overall system is exponentially stable if the product of the two Lipschitz constants satisfies the inequality:*

$$L_1 \cdot L_2 < \frac{\alpha_3}{\alpha_4} \cdot \frac{\beta_3}{\beta_4}. \tag{5.14}$$

*Proof.* The proof is very similar to that of Theorem 5.1. We consider the candidate Lyapunov function $V = V^I + \mu V^O$ for the overall system. Then we have:

$$\begin{aligned} \dot{V} = \dot{V}^I + \mu \dot{V}^O &\le -\beta_3\|e^I\|^2 + \beta_4 L_2\|e^I\|\|e^O\| - \mu\alpha_3\|e^O\|^2 + \mu\alpha_4 L_1\|e^O\|\|e^I\| \\ &= -(\|e^I\|, \|e^O\|)Q(\|e^I\|, \|e^O\|)^T \end{aligned}$$

where the matrix $Q \in \mathbb{R}^{2\times 2}$ is:

$$Q = \begin{pmatrix} \beta_3 & -\frac{1}{2}(\beta_4 L_2 + \mu\alpha_4 L_1) \\ -\frac{1}{2}(\beta_4 L_2 + \mu\alpha_4 L_1) & \mu\alpha_3 \end{pmatrix}.$$

$Q$ is positive definite if and only if $\det(Q) > 0$. That is, there exists $\mu > 0$ such that:

$$-\alpha_4^2 L_1^2 \mu^2 + (4\beta_3\alpha_3 - 2\beta_4 L_2 \alpha_4 L_1)\mu - \beta_4^2 L_2^2 > 0.$$

This is true if and only if the discriminant of the quadratic function of $\mu$ on the left hand side is positive which yields: $L_1 \cdot L_2 < \frac{\alpha_3}{\alpha_4} \cdot \frac{\beta_3}{\beta_4}$. □

This theorem states a very interesting fact about the system (5.13): heuristically, $\alpha_3$ and $\beta_3$ are proportional to the convergence rates of the outer and inner systems respectively,[2] hence the stability of the perturbed systems requires *only* that the *product* of the Lipschitz constants of the perturbation terms is less than the *product* of the two convergence rates, regardless of the rate of each individual system.

**Comments 5.3.** *The stability of a similar model of the overall closed-loop system has been studied before in [24], however, no explicit conditions are provided under which a $\mu$ exists such that the overall system is stable. Here, Theorems 5.1 and 5.2 give more detailed and useful results in characterizing the properties of the closed-loop system.*

[2]A more precise estimates of the convergences rates are given by $\frac{\alpha_3}{2\alpha_2}$ and $\frac{\beta_3}{2\beta_2}$.

Although we have established the conditions for the system (5.13) to be exponentially stable, estimates of its Lyapunov constants indeed depend on $L_1, L_2$ and all the Lyapunov constants of the inner and outer systems. These constants can be optimized by maximizing the smaller eigenvalue of $Q$ with respect to $\mu$. We here omit the detail and carry on the analysis by assuming that the system (5.13) is exponentially stable and its Lyapunov constants are denoted by $\gamma_1, \gamma_2, \gamma_3, \gamma_4 > 0$. We now want to estimate the effect of the non-vanishing error term $h$ on the performance of the closed-loop system (5.8). In general, we can no longer expect asymptotic stability when a non-vanishing perturbation is introduced. However, according to [9], we can still have good estimates of a bound on the tracking error and the rate of convergence outside this bound.

**Proposition 5.4.** *Assume that the system (5.13) has the Lyapunov constants* $\{\gamma_i\}_{i=1}^{4}$. *For the closed-loop system (5.8), if* $\|h(e^I, e^O, t)\| \leq \delta < \frac{\gamma_3}{2\gamma_4}\sqrt{\frac{\gamma_1}{\gamma_2}}$, *then the tracking error of the overall system is bounded by* $b = \frac{2\gamma_4}{\gamma_3}\sqrt{\frac{\gamma_2}{\gamma_1}}\delta$, *and, outside this bound, the error exponentially decreases with a rate larger than* $\lambda = \frac{\gamma_3}{4\gamma_2}$.

The control parameters $K_v$ and $K_p$ can be adjusted so as to minimize the error bound $b$. For the helicopter model, the error term $h(e^I, e^O, t)$ is usually extremely small, as is $\delta$. We can also choose the control parameters such that the inner and outer systems have very fast rates of convergence, hence a large $\gamma_3$. Consequently, the error bound $b$ is very small, and usually barely noticeable in simulations and experiments, as we will soon see.

# Chapter 6

# Vision in the Control Loop

In this chapter, we discuss how the discrete and differential motion estimation algorithms described in section 3 are used in the control loop for landing a UAV onto a landing pad with a known geometry.

## 6.1 Disambiguation of Motion Estimates

We assume that the image of the landing pad taken from the *desired* landing configuration are given. The feature points on the landing pad are assumed to be in general configuration (they could for example be corners on the typical "H" pattern). Without loss of generality, suppose $(p_0, I) \in SE(3)$ is the configuration of the desired camera frame, and $d_0 = -n_F^T p_0 > 0$ is the desired distance of the camera to the landing plane with known surface normal $n_F \in \mathbb{R}^3$.

**Proposition 6.1.** *Suppose $A = (R + \frac{1}{d} p n^T) \in \mathbb{R}^{3\times 3}$ is the planar essential matrix associated with two camera frames relative to a plane. If $d_0 > 0$ is the distance from the first camera to the plane, then the distance of the second camera to the plane is given by $d = d_0 / \det(A)$.*

*Proof.* Suppose the configuration of the second camera frame is $(p_1, R_1) \in SE(3)$. Then $d_0 = -n_F^T p_0$, $d = -n_F^T p_1$ are the distances from the first and second cameras to the plane.

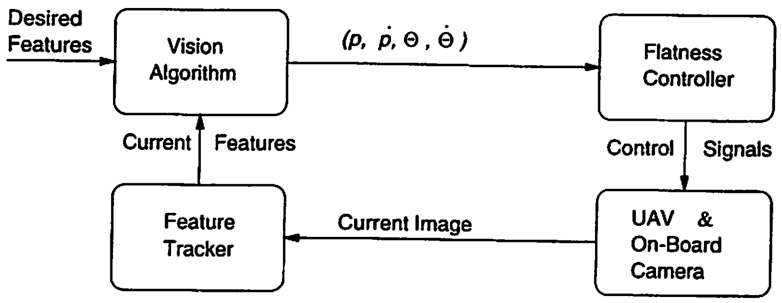


Figure 6.1: Block diagram of vision in control loop.

Since $n_F = Rn$, we have $AR^T = (I + \frac{1}{d}pn_F^T)$, hence the eigenvalues of $AR^T$ are $\{\lambda, 1, 1\}$ where $\lambda = 1 + \frac{1}{d}n_F^T p$. But $p^T n_F = p_1^T n_F - p_0^T n_F = -d + d_0$. Using $\det(A) = \det(AR^T) = \lambda$, it is direct to check that $\det(A) = d_0/d$. □

The knowledge of $n_F$ allows us to disambiguate the pair of solutions discussed in Theorem 3.9 by taking the one that minimizes $\|n_{\text{est}} - R_{\text{est}}^T n_F\|$, where $n_{\text{est}}$ is the vision estimated surface normal, and $R_{\text{est}}$ is the estimated rotation matrix according to the discrete algorithm. Also, the knowledge of $d_0$ allows to find $d$ according to Proposition 6.1, which solves the scale ambiguity in $p/d$ in the discrete case and $v/d$ in the differential case.

The vision algorithms described above generate estimates of $\{p, R, v, \omega\}$. However, to compute the control signals we need estimates of $\{p, \dot{p}, \Theta, \dot{\Theta}\}$. Note that given $R \in SO(3)$, the $ZYX$ Euler angles (away from the singularity) can be recovered by:

$$\begin{cases} \theta &= \operatorname{atan2}(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}) \\ \phi &= \operatorname{atan2}(r_{32}/\cos\theta, r_{33}/\cos\theta) \\ \psi &= \operatorname{atan2}(r_{21}/\cos\theta, r_{11}/\cos\theta) \end{cases} \tag{6.1}$$

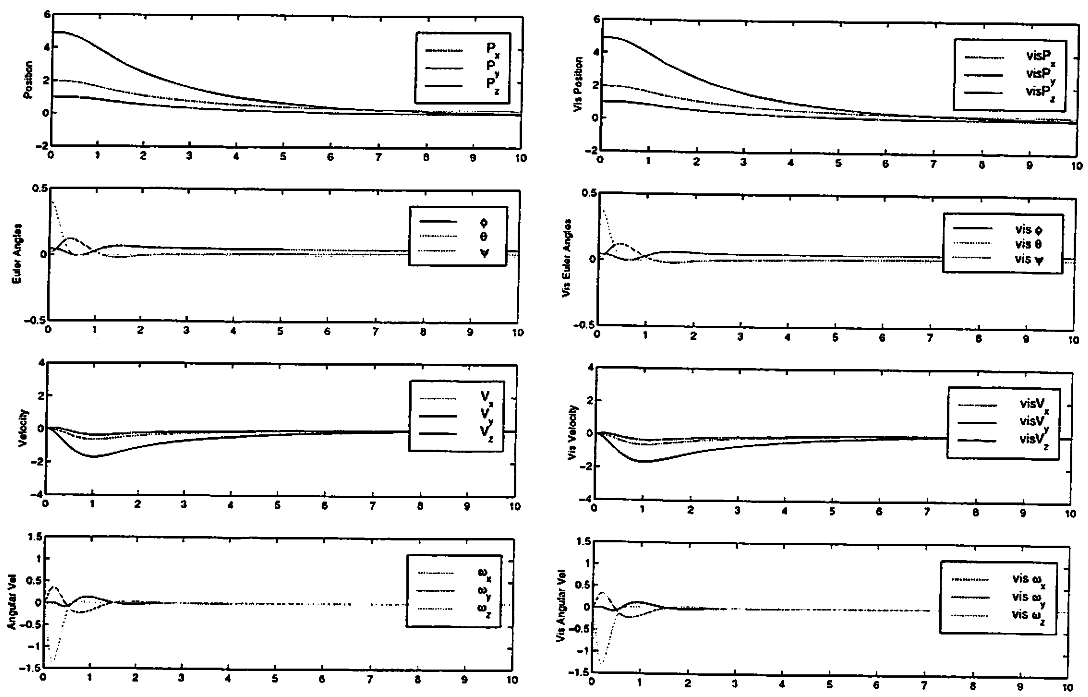
where $r_{ij}$ is the entry of the $i-$th row and $j-$th column of $R$. Thus, we can recover $\{\Theta, \dot{\Theta}\}$ from $\{R, \omega\}$ by applying equations (6.1), (5.1) and $\dot{\Theta} = \Psi\omega$. We can recover $\dot{p}$ using the estimates $\{v, R\}$ through $\dot{p} = Rv$. The closed-loop system configuration is depicted in Figure 6.1. For the estimate of $T_M$ one needs $\ddot{p}$ as in equation (5.6), which can be measured by accelerometers that give $a = R^T\ddot{p}$

## 6.2 Simulation Results for the Closed-Loop System

We present the simulation results of the proposed vision based landing scheme. In these simulations, we apply the proposed controller for the full dynamic model of the UAV. We add Gaussian noise of different levels of standard deviation (in pixel units) to the correspondences and image velocities, and perform the discrete and differential motion estimation algorithms based on the noisy data. In Figures 6.2 and 6.3 we present the simulation results for image measurement noise levels of $0, 1, 2, 4$ pixels standard deviation in both the image correspondences and the image velocities.

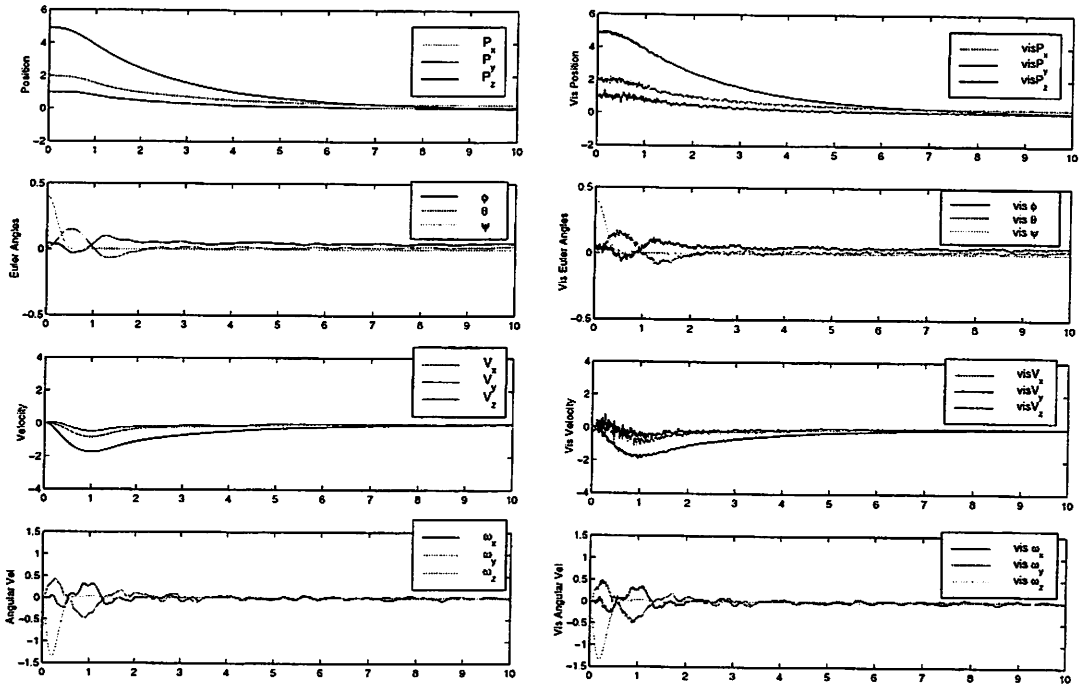
In these simulations, the initial position is $p = (2, 1, 5)^T$ meters away from the desired landing configuration above the landing pad (the origin), the initial orientation is $(\theta, \phi, \psi)^T = (0, 0, 0.4)^T$ radians. The dominant poles of the outer loop controller are placed at $-2, -.45$. The inner loop attitude controller is designed based on feedback linearization [10], and it has the form $\Theta^{(3)} = V_\Theta$, where $V_\Theta$ is designed as three decoupled pole-placement controllers with poles located at $-10$ and $-7 \pm 7.1414i$ for each controller. The main rotor thrust is controlled based on dynamic inversion and the pole is placed at $-5$.

Since the origin of the closed-loop system is exponentially stable, it is robust to relatively large levels of noise. As we see, the controller performs very well at a noise level of 1 pixel standard deviation, which is the accuracy of most state-of-the-art feature-tracking techniques [1], and remains stable at a large noise level of 4 pixel standard deviation. Due to the gain margin in the controller, the closed loop system is also robust to possible modeling errors which are omitted, such as the camera calibration.



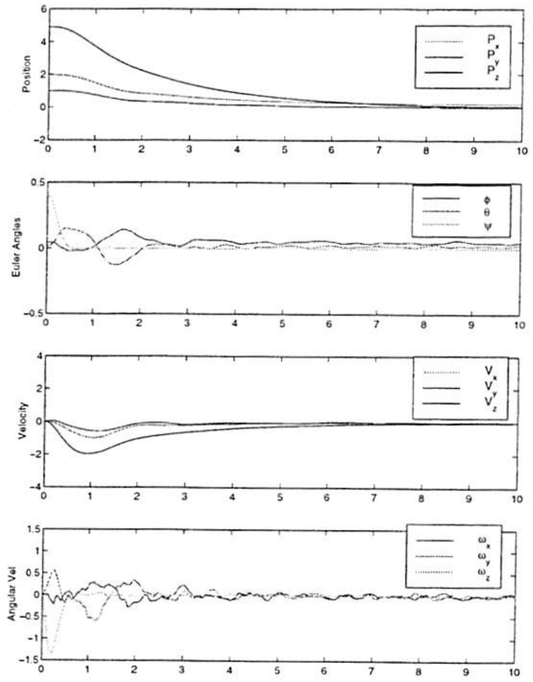
True motion. Vision estimated motion.

Figure 6.2: Closed-loop system landing simulation with 0 pixel noise.

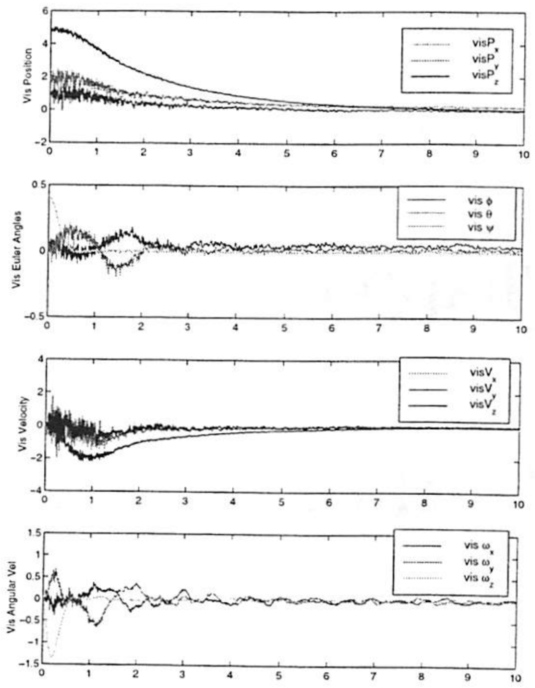


True motion. Vision estimated motion.

Figure 6.3: Closed-loop system landing simulation with 1 pixel noise.

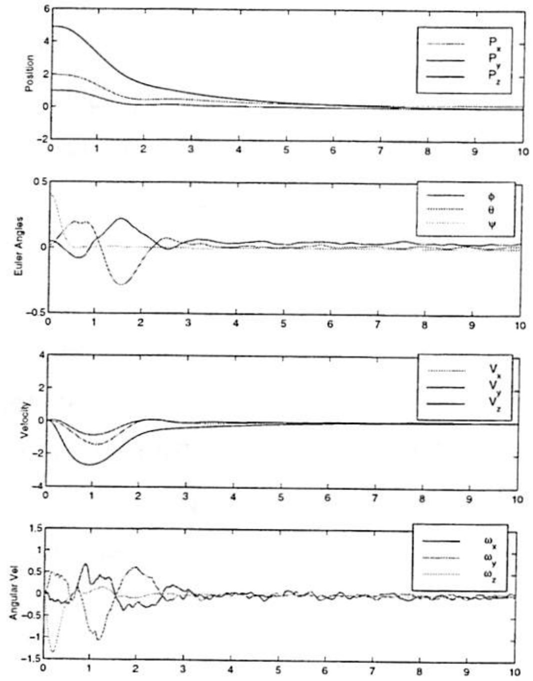


True motion.

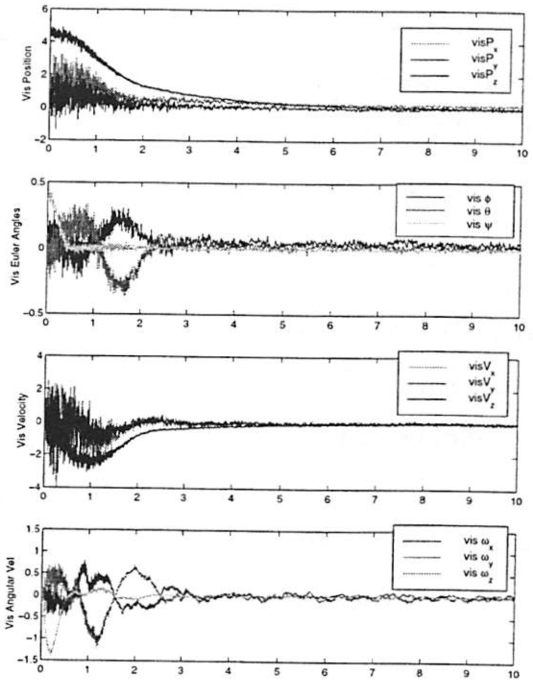


Vision estimated motion.

Figure 6.4: Closed-loop system landing simulation with 2 pixels noise.



True motion.



Vision estimated motion.

Figure 6.5: Closed-loop system landing simulation with 4 pixels noise.

# Chapter 7

# Conclusions

In this report we studied the problem of using computer vision as a feedback sensor to control the landing of an Unmanned Aerial Vehicle. We derived a novel geometric algorithm for estimating the camera angular and linear velocity relative to a planar scene, and presented a thorough performance evaluation of the algorithm. The algorithm was shown to be robust to noise in the image measurements and amenable to real-time implementation. We also proposed a nonlinear controller based on differential flatness for a full UAV dynamic model, and gave detailed conditions for stability of the overall closed loop system. Through extensive simulation, the vision based controller was shown to result in stable landing maneuvers for large noise levels.

We are currently developing a 3D visualization and simulation platform in order to evaluate image processing techniques, computer vision algorithms, and nonlinear control designs for the landing maneuver. Figure 7.1 shows screen shots of an early version of the program. We are incorporating a full dynamic model of a UAV, a model of a ship and various sea states, and photo-realistic 3D rendering in a unified simulation platform to capture all aspects of the vision based landing maneuver.

We are also implementing our vision algorithms and control design on a model helicopter as part of the BErkeley Aerial Robot (BEAR) project [7]. One of our UAVs is a Yamaha R-50 model helicopter, on which we have mounted computers, INS, GPS, and a vision system,

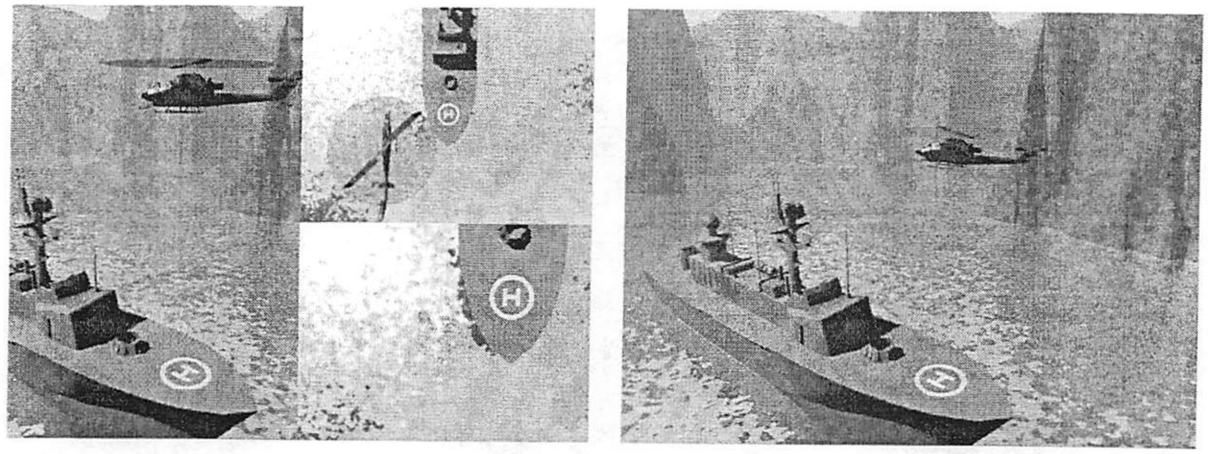


Figure 7.1: Showing screen shots of a 3D visualization of a UAV landing on a ship. Figure courtesy of Cory Sharp.

consisting of a camera, a real-time feature tracker board, and a Pentium II running Linux. Figure 7.2 shows one of the UC Berkeley UAVs on which we will implement the proposed landing scheme. A goal of the project is to perform an autonomous vision based landing of a BEAR UAV onto a pitching landing pad that simulates the motion of a ship in high seas. To this end we have developed a Stuart platform that simulates the motion of a ship. The landing deck system is controlled by a host computer which has a database of models of naval vessels and sea states. The landing deck moves according to the parameters of the sea state, such as amplitude, frequency, and a randomized phase, of the waves, the model of the ship, and the orientation of the ship relative to the direction of waves. Figure 7.3 shows the pitching deck that is under development for our landing experiments.

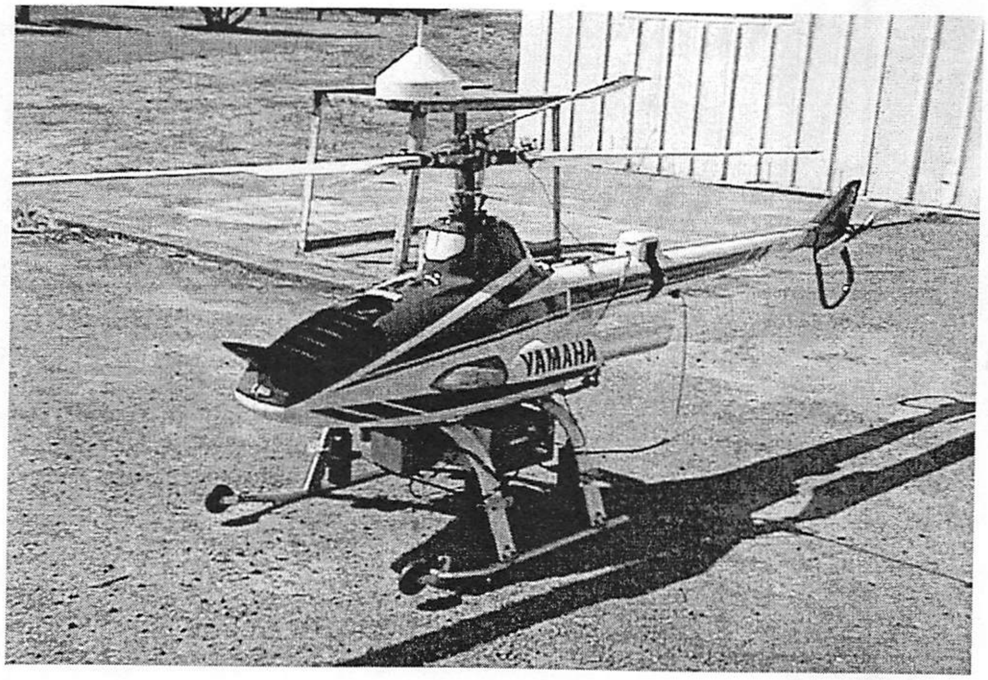


Figure 7.2: Showing a member of UC Berkeley UAV fleet. Figure courtesy of Hyunchul Shim.

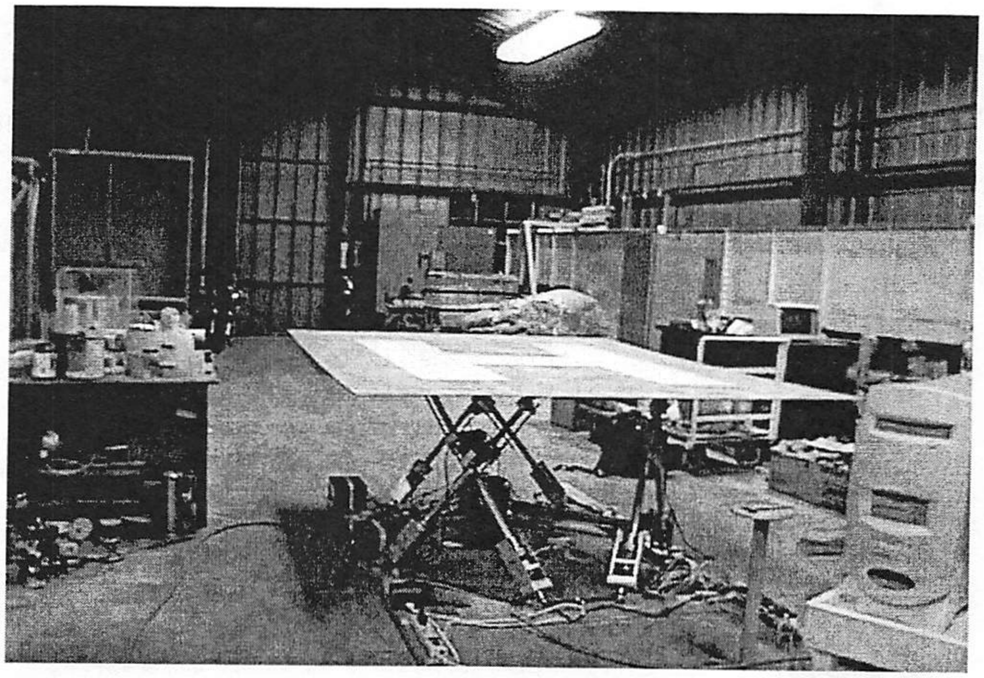


Figure 7.3: Showing the pitching landing deck. Figure courtesy of Tullio Celano III.

# Bibliography

[1] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *Int. Journal on Computer Vision*, 12(1):43–77, 1994.

[2] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313 - 326, June 1992.

[3] O.D. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508, 1988.

[4] O.D. Faugeras, F. Lustman, and G. Toscani. Motion and structure from motion from point and line matches. In *IEEE International Conference on Computer Vision*, pages 194–199, Washington DC, USA, 1987.

[5] M. Fliess. Aircraft control using flatness. In *Proceedings of Symposium on Control, Optimization and Supervision*, pages 194–9, Lille, France, July 1996.

[6] M. Fliess, J. Lévine, Ph. Martin, and P. Rouchon. Flatness and defect of nonlinear systems: introductory theory and applications. *Int. Journal of Control*, 61(6):1327–1361, 1995.

[7] BErkeley Aerial Robot (BEAR) Project Homepage. `http://robotics.eecs.berkeley.edu/bear`.

[8] K. Kanatani. Detecting the motion of a planar surface by line & surface integrals. In *Computer Vision, Graphics, and Image Processing*, volume 29, pages 13–22, 1985.

[9] H.K. Khalil. *Nonlinear Systems*. Prentice-Hall, 2nd edition, 1996.

[10] T.J. Koo and S. Sastry. Output tracking control design of a helicopter model based on approximate linearization. In *Proceedings of the 37th Conference on Decision and Control*, pages 3635–40, Tampa, Florida, December 1998.

[11] E. H. Lee, H. Shim, H. Park, and K. I. Lee. Design of hovering attitude controller for a model helicopter. In *Proceedings of Society of Instrument and Control Engineers*, pages 1385–1389, Tokyo, Japan, August 1993.

[12] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. In *Nature*, volume 293, pages 133–135, London, UK, 1981.

[13] H.C. Longuet-Higgins. The reconstruction of a plane surface from two perspective projections. In *Proceedings of Royal Society of London*, volume 227 of *B*, pages 399–410, 1986.

[14] Y. Ma, J. Košecká, and S.S. Sastry. Linear differential algorithm for motion recovery: A geometric approach. *International Journal of Computer Vision*, to appear.

[15] S. Maybank. *Theory of Reconstruction from Image Motion*. Springer Series in Information Sciences. Springer-Verlag, 1993.

[16] R.M. Murray, Z. Li, and S.S. Sastry. *A Mathematical Introduction to Robotic Manipulation.* CRC Press, 1993.

[17] Raymond W. Prouty. *Helicopter Performance, Stability, and Control.* Krieger Publishing Co., Inc., Boston, USA, 1995.

[18] S. Sastry. *Nonlinear Systems: Analysis, Stability and Control.* Springer-Verlag, 1999.

[19] F.R. Schell and E.D. Dickmanns. Autonomous landing of airplanes by dynamic machine vision. *Machine Vision and Applications*, 7:127–134, 1994.

[20] O. Shakernia, Y. Ma, T.J. Koo, J. Hespanha, and S.S. Sastry. Vision guided landing of an unmanned air vehicle. In *Proceedings of the 38th Conference on Decision & Control*, pages 4143–4148, Phoenix, Arizona, December 1999.

[21] O. Shakernia, Y. Ma, T.J. Koo, and S.S. Sastry. Landing an unmanned air vehicle: Vision based motion estimation and nonlinear control. *Asian Journal of Control*, 1(3):128–145, September 1999.

[22] M. Subbarao and A.M. Waxman. On the uniqueness of image flow solutions for planar surfaces in motion. *Computer Vision, Graphics, and Image Processing*, 36:208–228, 1986.

[23] G. Toscani and O.D. Faugeras. Structure and motion from two noisy perspective images. In *IEEE Int. Conference on Robotics & Automation*, pages 221–227, 1986.

[24] M. J. van Nieuwstadt and R. M. Murray. Outer flatness: Trajectory generation for a model helicopter. In *Proceedings of European Control Conference*, Brussels, Belgium, 1997.

[25] A. Waxman and S. Ullman. Surface structure and three-dimensional motion from image flow kinematics. *Int. J. Robotics Research*, 4(3):72–94, 1985.

[26] J. Weng, T.S. Huang, and N. Ahuja. *Motion and Structure from Image Sequences.* Springer-Verlag, 1993.

[27] S. Werner, S. Furst, D. Dickmanns, and E.D. Dickmanns. A vision-based multi-sensor machine perception system for autonomous aircraft landing approach. In *Proceedings of the SPIE - The International Society for Optical Engineering*, volume 2736, pages 54–63, Orlando, FL, USA, 1996.

[28] Z.F. Yang and W.H. Tsai. Using parallel line information for vision-based landmark location estimation and an application to automatic helicopter landing. *Robotics and Computer-Integrated Manufacturing*, 14(4):297–306, 1998.

# Appendix A

# Proof of Proposition 3.6

*Proof.* We will use the fact that a set of points in the plane are collinear if and only if the images of the points are collinear in the image plane [26]. This allows us to work with the images of features points on the plane.

For sufficiency, suppose there exists a set of four points in the plane such that no three are collinear. By contradiction, we will prove that the corresponding eight rows of $\mathbf{G}$ are linearly independent. Suppose the matrix:

$$\mathbf{G}^T = \begin{pmatrix} \mathbf{x}_1 & 0 & \mathbf{x}_2 & 0 & \mathbf{x}_3 & 0 & \mathbf{x}_4 & 0 \\ 0 & \mathbf{x}_1 & 0 & \mathbf{x}_2 & 0 & \mathbf{x}_3 & 0 & \mathbf{x}_4 \\ -x_1\mathbf{x}_1 & -y_1\mathbf{x}_1 & -x_2\mathbf{x}_2 & -y_2\mathbf{x}_2 & -x_3\mathbf{x}_3 & -y_3\mathbf{x}_3 & -x_4\mathbf{x}_4 & -y_4\mathbf{x}_4 \end{pmatrix} \in \mathbb{R}^{9\times 8} \quad (A.1)$$

has rank(G) $< 8$. Then there exists $\xi = (a_1, c_1, a_2, c_2, a_3, c_3, a_4, c_4)^T \in \mathbb{R}^8$ such than $\xi \neq 0$ and $\mathbf{G}^T\xi = 0$. Define $d_i = a_i x_i + c_i y_i$. Now let $\mathbf{a} = (a_1, a_2, a_3, a_4)^T$, $\mathbf{c} = (c_1, c_2, c_3, c_4)^T$, $\mathbf{d} = (d_1, d_2, d_3, d_4)^T$ and define $X = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4) \in \mathbb{R}^{3\times 4}$. With this notation, the condition $\mathbf{G}^T\xi = 0$, $\xi \neq 0$ implies $\mathbf{a} \neq 0$ or $\mathbf{c} \neq 0$ and $X\mathbf{a} = X\mathbf{c} = X\mathbf{d} = 0$.

Without loss of generality, take $\mathbf{a} \neq 0$. Since by the hypothesis, no three of $\mathbf{x}_i$ are collinear, each set of 3 columns of $X$ are linearly independent. Since $X\mathbf{a} = 0$, then if one component of $\mathbf{a}$ is zero, then we must have $\mathbf{a} = 0$. Hence $\mathbf{a} \neq 0$ implies $a_i \neq 0$ for $i = 1, \ldots, 4$. Since each set of 3 columns of $X$ are linearly independent $\dim(\ker(X)) \leq 1$ and $\exists \gamma, \delta \in \mathbb{R}$

such that $\mathbf{c} = \gamma\mathbf{a}$ and $\mathbf{d} = \delta\mathbf{a}$. This implies:

$$d_i = a_i x_i + c_i y_i = a_i x_i + \gamma a_i y_i = \delta a_i. \tag{A.2}$$

But since $a_i \neq 0$ for each $i$, we have $x_i + \gamma y_i = \delta$ which implies that all four image points $\mathbf{x}_i$ are collinear in the image plane, resulting in a contradiction.

For necessity, we first show that if all points are collinear, then rank(G) $\leq 5$. Let $u = (\alpha, \beta, 0) \in \mathbb{R}^3$ be the unit normal to the line in the image plane containing the image points $\mathbf{x}_i, i = 1, \ldots, m$. That is $\mathbf{x}_i^T u = 0$ for $i = 1, \ldots, m$. Define four vectors in $\mathbb{R}^9$ by:

$$h_1 = \begin{pmatrix} u \\ 0 \\ 0 \end{pmatrix}, \quad h_2 = \begin{pmatrix} 0 \\ u \\ 0 \end{pmatrix}, \quad h_3 = \begin{pmatrix} 0 \\ 0 \\ u \end{pmatrix}, \quad h_4 = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} \in \mathbb{R}^9 \tag{A.3}$$

where $(e_1, e_2, e_3) = I_{3\times3}$. Using $u^T u = 1$ and $e_3^T u = 0$, it is direct to check that for $H = (h_1, h_2, h_3, h_4) \in \mathbb{R}^{9\times4}$, $\det(H^T H) = 2$ and hence rank$(H) = 4$. From the structure of $\mathbf{G}$ in equation (A.1) is is clear that $\mathbf{G}h_i = 0$ for $i = 1, \ldots, 4$. Then dim(ker(G)) $\geq 4$ and hence rank(G) $\leq 9 - 4 = 5$.

Now suppose condition of the proposition is not satisfied. The claim is trivially proved if the number of image points is less than 3. Suppose there are more than 4 image points, not all collinear, and for each set of four points at least 3 are collinear. Without loss of generality, suppose $\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ lie on a line (call this the common line), and $\mathbf{x}_1$ does not lie on the common line. By induction, we prove that all $\mathbf{x}_i$'s for $i \geq 4$ lie on the common line. Suppose $\mathbf{x}_i$ lies on the common line for some $i \geq 4$ and $\mathbf{x}_{i+1}$ does not. Choose two points out of $\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_i$ such that they do not lie at the intersection of the common line and the line connecting $\mathbf{x}_1, \mathbf{x}_{i+1}$. Call these points $\mathbf{x}_j, \mathbf{x}_k$. Then the four points $\mathbf{x}_1, \mathbf{x}_j, \mathbf{x}_k, \mathbf{x}_{i+1}$ are in a general configuration. This is a contradiction, and hence $\mathbf{x}_{i+1}$ lies on the common line. Since all image points lie on a single line expect for $\mathbf{x}_1$, then rank(G) $\leq 5 + 2 = 7$. □

# Appendix B

# UAV System Parameters

All variables except for the state variables and inputs are numeric constants, which can be obtained by measurements and experiments. The followings are the values of the constants:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $I_x$ | $=$ | 0.142413 | $I_y$ | $=$ | 0.271256 | $I_z$ | $=$ | 0.271492 |
| $l_M$ | $=$ | $-0.015$ | $y_M$ | $=$ | 0 | $h_M$ | $=$ | 0.2943 |
| $h_T$ | $=$ | 0.1154 | $l_T$ | $=$ | 0.8715 | $m$ | $=$ | 4.9 |
| $C_M^Q$ | $=$ | 0.004452 | $D_M^Q$ | $=$ | 0.6304 | $\frac{\partial R_M}{\partial b_{1s}}$ | $=$ | 25.23 |
| $C_T^Q$ | $=$ | 0.005066 | $D_T^Q$ | $=$ | 0.008488 | $\frac{\partial M_M}{\partial a_{1s}}$ | $=$ | 25.23 |
| $c_{M1}$ | $=$ | 6.4578 | $c_{M3}$ | $=$ | 100.3752 | $c_{T1}$ | $=$ | 0.1837 |
| $c_{T3}$ | $=$ | 0.1545 | | | | | | |

The operation regions in radian for $a_{1s}, b_{1s}$ and newton for $T_M, T_T$ are: $|a_{1s}| \leq 0.4363$, $|b_{1s}| \leq 0.3491$, $-20.86 \leq T_M \leq 69.48$, $-5.26 \leq T_T \leq 5.26$.