Aggressive Landing Maneuvers for Unmanned Aerial Vehicles

by

Selcuk Bayraktar

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2006

Author .....
Department of Aeronautics and Astronautics
February 3, 2006

Certified by..... .....
Eric Feron
Visiting Professor
Thesis Supervisor

Accepted by..
Jaime Peraire
Professor of Aeronautics and Astronautics
Chairman, Department Committee on Graduate Students

Aggressive Landing Maneuvers for Unmanned Aerial Vehicles

by

Selcuk Bayraktar

Submitted to the Department of Aeronautics and Astronautics
on February 10, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

VTOL (Vertical Take Off and Landing) vehicle landing is considered to be a critically difficult task for both land, marine, and urban operations. This thesis describes one possible control approach to enable landing of unmanned aircraft systems at all attitudes, including against walls and ceilings as a way to considerably enhance the operational capability of these vehicles. The features of the research include a novel approach to trajectory tracking, whereby the primary system outputs to be tracked are smoothly scheduled according to the state of the vehicle relative to its landing area. The proposed approach is illustrated with several experiments using a low-cost three-degree-of-freedom helicopter. We also include the design details of a testbed for the demonstration of the application of our research endeavor. The testbed is a model helicopter UAV platform that has indoor and outdoor aggressive flight capability.

Thesis Supervisor: Eric Feron
Title: Visiting Professor

Acknowledgments

First and foremost, I would like to extend my infinite thanks to my Dear LORD, who is the most gracious, and the most merciful. Few words can express the gratitude, I feel from the depth of my heart, for His countless bounties.

I would like continue by thanking my advisor Prof. Eric Feron for his mentorship, his sincerity and his friendship. He means much more than just an advisor to me.

It has been a great pleasure for me to meet my dear lab mates, and even certainly a greater one to spend countless hours with them in the lab. In alphabetical order:

Shan-Yuan Ho aka HoHo for the great coffee and chat sessions after midnight. Bernard Mettler for his advice, Navid Sabbaghi for our deep conversations, Tom Schouwenaars for having such a pure heart, Olivier Toupet for all the great times of friendship, Glenn Tournier for being such a helpful and a warm face.

The MIT experience has been a personal transformation for me, mainly because of my dear friends and their beautiful example. Very special thanks go to my friend Abdulbasier Aziz, who is dear to me like my own brother, and the members of the MITMSA community who affected me deeply.

There is many other names I would like to thank. In order not to be unfair to anyone by forgetting their names, I am rather choosing to thank them in person.

Lastly, I would like to thank my dear Mother and my dear Father for their continuous love, support and sacrifice, none of which I can pay back.

Selcuk Bayraktar

MIT 10-Feb-2006

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Vertical Take-Off and Landing vehicles represent a fundamental component of Naval, Air and Ground support operations, because they are able to take-off and land in confined spaces and can therefore be deployed quite rapidly in unequipped or hostile areas. These characteristics also yield multiple uses in civilian operations, including police, firefighting, search and rescue, newscasting and other operations. Among the tasks associated with VTOL vehicle operations, take-off and landing certainly stand among the most critical ones, since the vehicles must operate and establish physical contact with the platform upon which they land. Unlike their fixed-wing counterparts, which must rely on a paved or unpaved runway system and therefore require a nontrivial infrastructure, the freedom and flexibility offered by VTOL operations are often leveraged together with the pilot expertise to perform landing operations in a wide variety of environments. However, the recent advent of Unmanned Aerial Vehicles has removed the pilot from the immediate controls of the vehicle, revealing many of the difficulties associated with VTOL landing operations. During landing, several problems must be considered, including those of (i) properly sensing the environment at or near the landing platform as addressed in [14] and (ii) maneuvering the vehicle in that environment in such a way that landing is possible and safe. In this thesis, we assume the environment to be known and available to the vehicle in real-time, and we concentrate on the task of landing the vehicle. In an attempt to push the state-of-the-art, our effort deliberately emphasizes all-attitude landing, i.e,

landing of vehicles on flat, inclined or vertical surfaces and possibly inverted landings as well. Following an argument adopted in prior research on aerobatic flight [10], our motivation for such an effort is twofold:

- Demonstrate that landings are possible at unusual attitudes, thereby extending the range of conditions over which hover-capable vehicles, small and big, might be able to stop and resume flight.

- Indirectly demonstrate the safety and robustness of more traditional landings, by displaying a much larger range of landing capabilities for the machine.

When concerned with landing operations, several factors need to be accounted for, including the fact that many VTOL vehicles are *underactuated*, that is, the number of control variables available to them (pitch/roll cyclics, collective and tail rotor in the case of helicopters) is smaller than the number of possible outputs of interest, which includes all six degrees of freedom of the vehicle (which evolves in $SE(3)$ if the vehicle is seen as a rigid body). When the vehicle is limited to three degrees of freedom (such as the system described and used later in this thesis), the number of actuation mechanisms is often limited to two, enough to manage the vehicle attitude or its position, but not both at the same time. However, it is often possible to "focus" attention onto subsets of these outputs: For example, while vehicle position (e.g deviations from the glideslope) might be of primary interest during approach, vehicle attitude (or relative attitude with respect to the landing site) may become more important when the vehicle is about to land. Similar statements hold true about many other vehicles, with car driving and maneuvering a familiar everyday example.

In this thesis, based on an aggressive landing scenario, we describe a possible control approach towards managing system outputs and corresponding control strategies for underactuated systems. Our approach provides smooth control law transition strategies when the "critical outputs" of the system vary as a function of state-dependent constraints, such as vehicle proximity to obstacles. The basic principle of our approach is, for a desired nominal (and feasible) trajectory to be followed by the vehicle, given a number n of inputs and $m > n$ of outputs, to first design separate

compensators for a given family of subsets of n outputs. We then apply a weighted least-squares scheme to come up with a control action that properly trades off the relative importance of these outputs, by correspondingly weighting the corresponding control actions.

Finally we conclude with the design details of a testbed for the demonstration of the application of the research endeavor. The testbed is a model helicopter UAV platform and has indoor and outdoor aggressive flight capability.

Chapter 2

Method

2.1 Overview of the Proposed Approach

We assume that we are given a trajectory pair $(x(t), u(t))$ which is compatible with the vehicle dynamics, which fall within the broad form of control affine systems:

$$\dot{x} = f(x) + g(x)u \text{ where } x \in \mathbb{R}^n,\ u(t) \in \{\text{Actuation Set}\} \subset \mathbb{R}^m \tag{2.1}$$

Together with the trajectory, we also assume the existence of a penalty function on the system's most important outputs. We assume this penalty function to be a function of the *states* of the system. The penalty function could be interpreted either as a measure of convergence priority or tracking tolerance of the selected outputs in different regions of interest.

Consider the following example: Assume we would like to land our helicopter UAV vertically on a wall or conventionally (horizontally) on a ship deck. In both cases we achieve this by a sequence maneuvers: Sloped descent followed by a flare maneuver in the case of ship deck landing, or followed by a pull up maneuver in the case of landing on a vertical wall. In both cases the term "maneuver" refers to a trajectory which is not a trivial element of the equilibrium manifold of the equations of motion with respect to spatial position or spatial velocity. Thus in a conventional ship landing case, approaching the ship with a configuration very close to hover and then descending

vertically and slowly would not be referred to as a "maneuver" since the vehicle would almost be always close to trimmed conditions. Landing procedures based on keeping the vehicle in the state of "quasi-equilibrium" are standard practice and are not the object of our research. To motivate the necessity of the penalty function or, equivalently, a tracking tolerance function, consider the longitudinal dynamics of a vehicle when performing vertical and horizontal landings. During the approach phase, we do not care about the vehicle's attitude, but rather focus on altitude and forward position deviations from the nominal trajectory. During final flare or pull-up maneuver however, we require tighter tracking on the pitch angle.

The change of "important outputs" is unavoidable when landing on a vertical wall, since then vertical vehicle position is uncontrollable when the vehicle is close to landing. So in that sense and in the case of an underactuated and aerobatic vehicle, changing weight on the relative importance of outputs becomes necessary. We now detail our approach.

2.1.1 Generic Vehicle longitudinal dynamics

In the rest of this thesis, we consider only planar, longitudinal dynamics of the helicopter and fixed wing UAVs derived in the coordinate frame shown in figure 2-1, and the equations of motion given by 2.2 are essentially equivalent for our purposes.

$$\begin{aligned} \ddot{x} &= \tfrac{1}{m}u_1 \sin\theta + n_x(\mathbf{x}) \\ \ddot{z} &= \tfrac{1}{m}u_1 \cos\theta - g + n_y(\mathbf{x}) \\ \ddot{\theta} &= \tfrac{1}{j}u_2 + n_\theta(\mathbf{x}) \end{aligned} \tag{2.2}$$

In these equations, the boldface $\mathbf{x} \in \mathbb{R}^6$ is the state vector $\mathbf{x} = (x\ \dot{x}\ z\ \dot{z}\ \theta\ \dot{\theta})^T$ and $\mathbf{m}$ and $\mathbf{J}$ are the mass and the inertia of the UAV. The terms given by $n_x(\mathbf{x}), n_z(\mathbf{x}), n_\theta(\mathbf{x})$ represent forces that we will neglect in the rest of the thesis. We will extend our work to a broader class of vehicle dynamics as our research progresses.

The justification for our choice of the resulting simplified planar dynamics is as follows: For many miniature UAVs that our application is concerned with the thrust

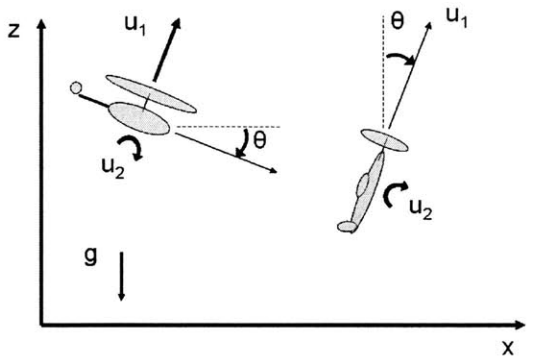


Figure 2-1: Planar UAV model: helicopter and fixed wing configuration.

to weight and the torque to inertia ratio are easily on the orders of $\approx 2:1$ for weight, and $\approx 5:1$ for inertia if not more. Henceforth, given the high bandwidth actuator authority on the one hand and our specific region of interest in the flight envelope being "aggressive region" for our particular application on the other hand, the aerodynamics effects like drag and lift on the vehicle nacelle are somewhat negligible.

2.1.2 Nonlinear Controller Design

We use the feedback linearization technique to derive our controllers . This is by now a standard technique to design tracking controllers. One very important and fundamental limitation of this and all other control design techniques is the fact that the number of outputs must be equal to the number of inputs, i.e the system must be "square" [18] to achieve perfect tracking. This is a significant limitation if the outputs of interest need to change during a trajectory execution. Let us motivate the problem and our devised method by designing the controllers for our application at hand : *Land vertically on a wall.*

We first derive the feedback linearizing controller using the dynamic extension algorithm mentioned in [19]. Choosing our outputs of interest to be (x, y) as in the beginning of landing we need to approach the landing pad. We differentiate our outputs until we get a locally smooth diffeomorphism back to our inputs. Applying the dynamic extension algorithm, we get the equation 2.3 and checking the singularity

of the mapping $\mathbf{T_m}$ in equation 2.4 we see that the we can only achieve feedback linearization with a dynamic compensator - specifically we must have a double integrator to get back to our input so our dynamics are augmented with the compensator dynamics [12].

$$\underbrace{\begin{bmatrix} x^{(4)} \\ z^{(4)} \end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix} \frac{\sin\theta}{m} & \frac{u_1\cos\theta}{Jm} \\ \frac{\cos\theta}{m} & -\frac{u_1\sin\theta}{Jm} \end{bmatrix}}_{T_m} \underbrace{\begin{bmatrix} \ddot{u}_1 \\ u_2 \end{bmatrix}}_{u} + \underbrace{\begin{bmatrix} \frac{2\dot{\theta}\cos\theta\dot{u}_1 - \dot{\theta}^2 u_1\sin\theta}{m} \\ \frac{-\dot{\theta}^2 u_1\cos\theta - 2\dot{\theta}\dot{u}_1\sin\theta}{m} \end{bmatrix}}_{b} \tag{2.3}$$

$$Det(T_m) = \frac{-u_1}{Jm^2} \quad Rank(T_m) = 2\ \forall u_1,\ u_1 \neq 0 \ \ assuming\ J, m > 0 \tag{2.4}$$

Since $\mathbf{T_m}$ is invertible as long as $u_1 \neq 0$, we can proceed to define a new input pair (v_x, v_z) through (2.5)

$$\mathbf{u} = \mathbf{T_m^{-1}}(\mathbf{v} - \mathbf{b}) \ \ where \ \ v = [v_x\ v_z]^T, \tag{2.5}$$

that decouples the (x, y) dynamics into two subsytems and puts them in the controllable canonical form as given by the equation (2.6). This also leaves the internal θ dynamics to be as shown in (2.7).

$$x^{(4)} = v_x \tag{2.6}$$

$$z^{(4)} = v_z$$

$$\theta^{(2)} = (1/u_1)(mv_x\cos\theta - 2\dot{\theta}\dot{u}_1 - mv_z\sin\theta) \tag{2.7}$$

Having designed our (x, y) controller with our linearizing inputs (v_x, v_z) we can easily design a trajectory tracker by essentially placing the poles of error dynamics wherever desired since the resulting two subsystems are decoupled, and both are controllable. One easy choice for a tracker would be as given in (2.8) below:

$$
\begin{aligned}
v_x &= x_d^{(4)} - 4\lambda_x(x^{(3)} - x_d^{(3)}) - 6\lambda_x^2(x^{(2)} - x_d^{(2)}) - 4\lambda_x^3(x^{(1)} - x_d^{(1)}) - \lambda_x^4(x - x_d) \\
v_z &= z_d^{(4)} - 4\lambda_z(z^{(3)} - z_d^{(3)}) - 6\lambda_z^2(z^{(2)} - z_d^{(2)}) - 4\lambda_z^3(z^{(1)} - z_d^{(1)}) - \lambda_z^4(z - z_d) \quad (2.8) \\
&\quad where \;\; \lambda_x, \lambda_z > 0
\end{aligned}
$$

So the total relative degree of this system is 8, whereas the total dimension of the original system was 6. Since the dynamic extension procedure augmented the system with the compensator dynamics, namely the double integrator on input u_1, our total system dimension is $6+2 = 8$. Since this is equal to the total relative degree of the system, there is no observable internal dynamics with this choice of outputs as suggested by [19]. So with the aid of this controller (2.8), we are guaranteed asymptotic and exponential convergence to any sufficiently smooth trajectory in our output space $x(t), y(t) \in C^4$, assuming also that the desired trajectory does not violate our actuation constraints. Once the system reaches the feasible trajectory, and if the system dynamics are exact, the system is guaranteed to stay on the trajectory (in the absence of perturbations). $(if \;\exists\; t_c \;\; s.t.\; \mathbf{e}(t_c) = 0 \;\; \Rightarrow \mathbf{e}(t) = 0 \;\forall t \geq t_c)$. However, if we drift away from the reference trajectory, we have no control over the θ-pitch axes convergence to the nominal trajectory. All we know *a priori* is exponential convergence in the error dynamics of (x, y) output pair.

In order to explain the significance of controlling the convergence behaviour, consider the following scenario: Imagine we have a feasible reference trajectory for landing on a vertical wall. Assume we begin on the reference trajectory using our (x, y) feedback linearization controller. At the final phase of the landing, assume that the vehicle drifts along the z axis (altitude), due to wind disturbance . Since the vehicle is almost vertical, the controller will try to generate large torque inputs to pitch the vehicle (θ axis) in order to generate the necessary exponential convergence in both cases. This can be seen by looking at the mapping from (v_x, v_z) to input pair (u_1, u_2). We can also see this intuitively from the equations of motion that there is no way of instantenously generating forces in the directions we require in such configurations.

Such pitching motion is undesirable since the vehicle is very close to the landing pad it could easily result the vehicle to hit its tail or its skids and crash.

One way around this problem would be to design other controllers with different outputs of interest and switch controllers whenever the tracking or the convergence behaviour in the specific outputs became more important. The natural choice of output pairs other than (x, y) would be the (x, θ) and (z, θ) pairs. So proceeding as before we can easily generate the input-output linearizing controllers as before. Please note that this procedure will render the internal dynamics unobservable in both cases. Namely the behavior of the states $(z\ z^{(1)})^T$ in the former and $(x\ x^{(1)})^T$ in the latter case will totally be disregarded by the controller in favor of achieving perfect tracking in the outputs of interest. Following a simple analysis, one can easily see that even for trivial constant trajectories the internal dynamics might easily become unstable.

2.1.3 Penalty Based Controller

In our quest to achieve tracking and convergence to reference trajectory without totally sacrificing one of the outputs we propose the following approach. We define our output to be $(x\ z\ \theta)^T$. Proceeding as before with the dynamic extension algorithm for the output pair (x, y) and augmenting this with the θ pitch dynamics, our equations now have the form:

$$\underbrace{\begin{bmatrix} x^{(4)} \\ z^{(4)} \\ \theta^{(2)} \end{bmatrix}}_{\dot{\hat{x}}} = \underbrace{\begin{bmatrix} \mathbf{T_m} \\ -- \\ 0\ 1 \end{bmatrix}}_{\hat{T}_m} \underbrace{\begin{bmatrix} \ddot{u}_1 \\ u_2 \end{bmatrix}}_{u} + \underbrace{\begin{bmatrix} \mathbf{b} \\ - \\ 0 \end{bmatrix}}_{\hat{b}} \tag{2.9}$$

We see that we could invert the dynamics or achieve desired convergence in all outputs with a controller, only when $(\mathbf{v} - \hat{\mathbf{b}})$ is in the column space of $\hat{\mathbf{T}}_\mathbf{m}$ where $\mathbf{v} = (v_x\ v_z\ v_\theta)^T$. Since this is not necessarily the case, we construct the following quadratic form or loosely an inner product and also define again loosely its induced

semi-norm as follows:

$$< a, b >= a^T W b \quad and \quad \|a\|^2 = a^T W a \ , \ W = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} \tag{2.10}$$

$$where\ a, b \in \Re^3 \ \ and\ d_1, d_2, d_3 \in \Re^+ \bigcup \{0\}\ only\ one\ of\ \ d_1, d_2, d_3 \ \ can\ be\ 0$$

We then take the weighted least squares solution to the problem at hand with respect to the semi-norm just defined to get the control input:

$$\mathbf{u} = (\hat{T}_m^T W \hat{T}_m)^{-1} \hat{T}_m^T W (\mathbf{v} - \hat{\mathbf{b}}) \ \ where\ \mathbf{u} = (\ddot{u}_1\ u_2)^T \ and\ \mathbf{v}, \mathbf{b}\ as\ defined\ before. \tag{2.11}$$

Figure 2-2 shows the available range of weight parameter selections for d_1, d_2 and d_3 values. The vertices represent extreme cases where one of the outputs is not effectively weighted. Therefore only the remaining two outputs are weighted for matching the desired value of the left hand side, in which case we get perfect linearization for those outputs. By varying the weight parameter selection in the convex hull of the 3 vertices, which represent $(\mathbf{x}, \mathbf{z})$ $(\mathbf{x}, \theta)$ $(\mathbf{z}, \theta)$ controllers respectively, we get the possible range of different weight penalties which lets us control the specific convergence of our outputs back on to the desired trajectory whenever the system is drifted off the trajectory.

Using the method outlined above and evaluating the value of $\mathbf{u}$ symbolically by plugging it back into the equations of motion (2.9), we get the equivalent dynamics of our system to be:

$$\begin{bmatrix} x^{(4)} \\ z^{(4)} \\ \theta^{(2)} \end{bmatrix} = \begin{bmatrix} \frac{d_1 v_x (d_3 m^2 + 2 d_2 u_1^2 - d_3 m^2 \cos(2\theta)) + 4 d_2 d_3 m \dot{\theta} \cos\theta \dot{u}_1 + 2 d_2 d_3 \cos\theta (u_1 v_\theta + m v_z \sin\theta)}{D} \\ \frac{d_2 v_z (d_3 m^2 + 2 d_1 u_1^2 + d_3 m^2 \cos(2\theta)) + 2 d_1 d_3 m (-u_1 v_\theta + m v_x \cos\theta - 2\dot{\theta}\dot{u}_1) \sin\theta}{D} \\ \frac{m(2 d_1 d_2 u_1 v_x \cos\theta + d_3 m v_\theta (d_1 + d_2 + (d_2 - d_1)\cos(2\theta))) - 2 d_1 d_2 u_1 (2\dot{\theta}\dot{u}_1 + m v_z \sin\theta)}{D} \end{bmatrix} \tag{2.12}$$

where

$$D = (d_1 + d_2) d_3 m^2 + 2 d_1 d_2 u_1^2 + (d_2 - d_1) d_3 m^2 \cos(2\theta)$$

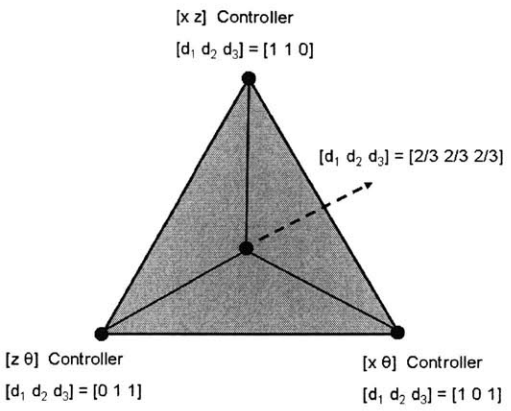


Figure 2-2: Range of parameters $(\mathbf{d_1}\ \mathbf{d_2}\ \mathbf{d_3})$.

Also evaluating the same procedure again for the values of $(d_1\, d_2\, d_3)$ at vertices of the triangle given in the figure 2-2 we get the equivalent linearized dynamics in equations (2.13) (2.14) (2.15) for the outputs (x, y), (x, θ) , and (z, θ) respectively.

$$\begin{bmatrix} x^{(4)} \\ z^{(4)} \\ \theta^{(2)} \end{bmatrix} = \begin{bmatrix} v_x \\ v_z \\ (1/u_1)(mv_x \cos\theta - 2\dot{\theta}\dot{u}_1 - mv_z \sin\theta \end{bmatrix} \tag{2.13}$$

$$\begin{bmatrix} x^{(4)} \\ z^{(4)} \\ \theta^{(2)} \end{bmatrix} = \begin{bmatrix} v_x \\ (1/m)\csc\theta(-u_1 v_\theta + mv_x \cos\theta - 2\dot{\theta}\dot{u}_1) \\ v_\theta \end{bmatrix} \tag{2.14}$$

$$\begin{bmatrix} x^{(4)} \\ z^{(4)} \\ \theta^{(2)} \end{bmatrix} = \begin{bmatrix} \sec\theta(1/m)(u_1 v_\theta + 2\dot{\theta}\dot{u}_1 + mv_z \sin\theta \\ v_z \\ v_\theta \end{bmatrix} \tag{2.15}$$

One more important point to note here is that there are some intrinsic singularities of our controllers for some regions of the system state space. They are mainly due to singularities of the affine mapping $\mathbf{T_m}$, $\mathbf{b}$ to the output space. They represent the regions of the state space where $\mathbf{T_m}$ loses its full rank, specifically corresponding to the regions where the actuation space has no projection over the output space.

They are given in (2.16) - (2.19) for the general values of the weights. (d_1 d_2 d_3) and specifically for the values of the weights at vertices of the triangle.

$$Det(T_m^T W T_m) = \frac{(d_1+d_2)\,d_3\,m^2+2\,d_1\,d_2\,u_1^2+(-d_1+d_2)\,d_3\,m^2\,\cos 2\theta)}{2\,J^2\,m^4} = 0 \quad (2.16)$$

$$Det(T_m^T W T_m) = \frac{u_1^2}{J^2 m^4} = 0 \quad (2.17)$$

$$Det(T_m^T W T_m) = \frac{(\sin\theta)^2}{J^2 m^2} = 0 \quad (2.18)$$

$$Det(T_m^T W T_m) = \frac{(\cos\theta)^2}{J^2 m^2} = 0 \quad (2.19)$$

2.1.4 Stability Verification and Tracking Performance

We have not formally verified the stability and performance of the proposed controller so far. Instead we extensively tested the controller with the planar model, including realistic actuator saturations in our simulations. We tried different combinations of the (d_1, d_2, d_3) in the convex hull of the triangle given by the figure. We also tested cases where the parameters were varying with time. The results were as expected: We obtained appropriate trade offs between the tracking error bounds (i.e in the interior of the triangle) versus perfect convergence on the chosen outputs (i.e at the vertices) depending on the weight parameters. Errors on all variables remain bounded if the weighting function keeps the "controller trade-off" sufficiently far away from the vertices.

Chapter 3

Experimental Setup and Experiments

3.1 Experimental Setup

3.1.1 3DOF Helicopter

Figure 3-1 shows our test setup which essentially emulates the flight of a reduced degrees of freedom (DOF) helicopter. Instead of the usual six DOF of a free-flying helicopter, the Quanser [17] only exhibits three: the pitch motion θ, the roll motion ϕ and the travel motion ψ. The 3DOF helicopter is mounted on a table top and its primary components are the main beam (mounted on a slip-ring), the twin rotor assembly and the counterweight. The system is actuated by two rotors driven each by an electric motor. The DC motors can provide either collective or cyclic voltage.

System Description

In order to derive the dynamic equations of the system, a coordinate system is used with its origin at the bearing and slip-ring assembly, with travel (ψ) being the circular motion of the main beam, (θ) being the vertical motion of the beam, and pitch (ϕ) being the motion of the rotor assembly. The collective ($T_{col} = T_L + T_R$) and cyclic ($T_{cyc} = T_L - T_R$) thrust serve as system inputs. The vertical motion is controlled

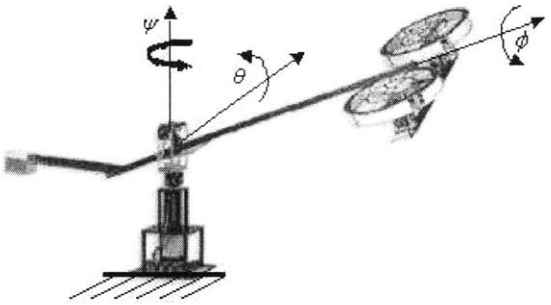


Figure 3-1: An illustration of the 3DOF helicopter.[13]

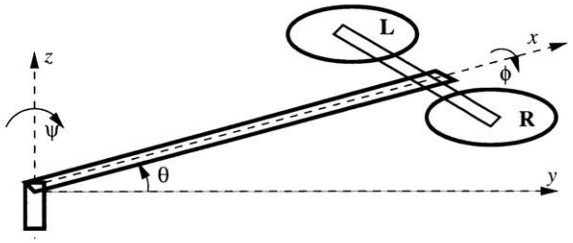


Figure 3-2: Schematic of the 3DOF helicopter.[13]

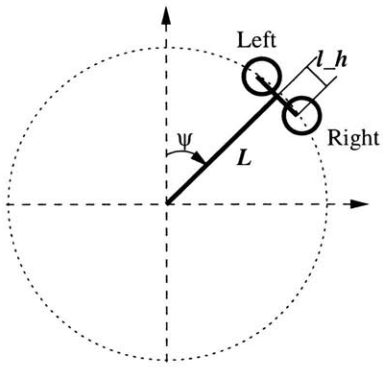


Figure 3-3: Top view. [13]

by collective thrust and is defined to be positive up from the horizontal position. The cyclic thrust produces a positive change in the pitch angle. If the pitch angle is non-zero, then the components of thrust will produce a torque around the travel axis. Positive roll results in positive change of travel angle. The schematics of helicopter are shown in Figures 3-2 and 3-3.

The equations of motion of 3DOF helicopter model are given in (3.2) as derived in [13] using Newton's second law to the rate of change of angular momentum. Let J_{xx}, J_{yy} and J_{zz} denote the corresponding principle moments of inertia. The moment of inertia matrix was calculated by disassembling the plant, measuring the dimensions and mass of each component and then drawing the system in CAD software. The inertia terms and other model parameters are shown in Table 3.1. As can be seen in the table the product of inertia terms are small and hence are neglected in the further

model derivation.

The equations of motion are as follows:

$$\begin{aligned} J_{zz}\ddot{\psi} &= (T_L + T_R)L\cos(\theta)\sin(\phi) - (T_L - T_R)l_h \sin(\theta)\sin(\phi) - \text{Drag} \\ J_{yy}\ddot{\theta} &= -Mgl_\theta \sin(\theta + \theta_0) + (T_L + T_R)L\cos(\phi) \\ J_{xx}\ddot{\phi} &= -mgl_\phi \sin(\phi) + (T_L - T_R)l_h \end{aligned} \tag{3.1}$$

- M is the total mass of the helicopter assembly
- m is the mass of the rotor assembly
- L is the length of the main beam from the slip-ring pivot to the rotor assembly
- l_h is the distance from the rotor pivot to each of the propellers
- $\text{Drag} = \frac{1}{2}\rho(\dot{\psi}L)^2(S_0 + S_0'\sin(\phi))L$
- S_0 and S_0' are the effective drag coefficients times the reference area and ρ is the density of air

Since the actual system inputs are motor voltages, a thrust to voltage relationship has been empirically determined and implemented using a lookup table. Figure 3-4 shows a relationship between the input voltage and the thrust for one of the motors. The thrust values determined empirically and the steady state thrust values were used in generating the graph. The dynamics of the DC motors were ignored as they are much faster compared to the rest of system dynamics.

Tracking Controllers

In order to apply te same procedure of Section 2.1.3 to the dynamics at hand, we used a simplified version of the dynamics in favor of removing the weak coupling of the T_{cyc} input to the travel ψ dynamics. The model with dominant terms is given as in (3.3).

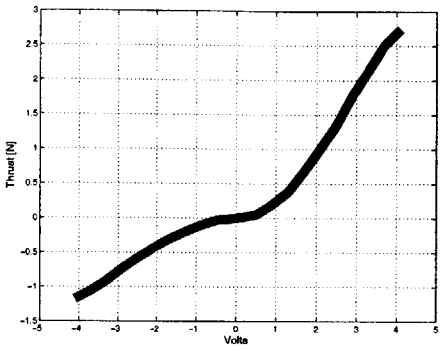


Figure 3-4: Motor Volt to Thrust Relationship.

Table 3.1: Parameter Values

Parameter	Value	Unit	Description
m	1.15	kg	mass of the rotor assembly
M	3.57	kg	mass of the whole setup
m_0	0.031	kg	effective mass at hover
L	0.66	m	length from pivot point to the heli body
l_h	0.177	m	length from pivot point to the rotor
J_{xx}	0.0344	kgm^2	moment of inertia about x-axis
J_{yy}	1.0197	kgm^2	moment of inertia about y-axis
J_{zz}	1.0349	kgm^2	moment of inertia about z-axis
J_{yz}	-0.0018	kgm^2	product of inertia
J_{zy}	-0.0018	kgm^2	product of inertia
R	0.1	m	radius of the rotor
g	9.8	m/s^2	gravitational constant
l_ϕ	0.004	m	length of pendulum for roll axis
l_θ	0.014	m	length of pendulum for pitch axis

$$\begin{aligned} \ddot{\psi} &= c_0 \cos(\theta)\sin(\phi)T_{col} \\ \ddot{\theta} &= c_1 \sin(\theta+\theta_0) + c_2 \cos(\phi)T_{col} \\ \ddot{\phi} &= c_3 T_{cyc} + c_4 \sin(\phi) \end{aligned} \tag{3.2}$$

$$c_0 = \frac{L}{J_{zz}} \quad c_1 = \frac{-Mgl_\theta}{J_{yy}} \quad c_2 = \frac{L}{J_{yy}} \quad c_3 = \frac{l_h}{J_{xx}} \quad c_4 = -\frac{mgl_\phi}{J_{xx}}$$

Applying the procedure of Section 2.1.3 we get the equivalent equations of the form given by (3.3).

$$\underbrace{\begin{bmatrix} \psi^{(4)} \\ \theta^{(4)} \\ \phi^{(2)} \end{bmatrix}}_{\dot{\hat{x}}} = \underbrace{\begin{bmatrix} c_0 \cos\theta \sin\phi & c_0 c_3 T_{col} \cos\phi \cos\theta \\ c_2 \cos\phi & -c_2 c_3 T_{col} \sin\phi \\ 0 & c_3 \end{bmatrix}}_{\hat{T}_m} \underbrace{\begin{bmatrix} \ddot{T}_{col} \\ T_{cyc} \end{bmatrix}}_{u} + \underbrace{\begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}}_{\hat{b}} \quad (3.3)$$

where

$$\begin{aligned}
\hat{b}_1 &= -\Big(c_0 \big(u_1 (\dot{\phi}^2 + \dot{\theta}^2 - c_4 \cos(\phi)) \cos(\theta) \sin(\phi) + u_1 (2\dot{\phi}\ \dot{\theta} \cos(\phi) \\
&\quad + \ddot{\theta} \sin(\phi)) \sin(\theta) + \dot{u}_1 (-2\dot{\phi} \cos(\phi) \cos(\theta) + 2\dot{\theta} \sin(\phi) \sin(\theta)) \big) \Big) \\
\hat{b}_2 &= -\big(c_2 \dot{\phi}^2 u_1 \cos(\phi) \big) + c_1 \ddot{\theta} \cos(\theta + \theta_0) - c_2 \sin(\phi) (2\dot{\phi} \dot{u}_1 + c_4 u_1 \sin(\phi)) \\
&\quad - c_1 \dot{\theta}^2 \sin(\theta + \theta_0) \\
\hat{b}_2 &= c_4 \sin(\phi)
\end{aligned}$$

3.2 Experiments

A series of aggressive vertical landing experiments with a set of different trajectories were performed with the 3DOF setup to test the validity of our approach. The trajectories used were feasible for the dynamics and they were experimentally obtained. approach. Figures 3-5 through 3-8 show one of the experimental results. In figure 3-5 the dashed lines represents the desired trajectory. In both figures 3-5 and 3-6 the solid line represents the actual trajectory. And the two bounding lines, upper and lower, represent equal penalty boundaries for the specific output. As can be seen from the figures as we increase the penalty on a specific output (which corresponds to narrowing of the equi-penalty boundary), the tracking error decreases as desired.

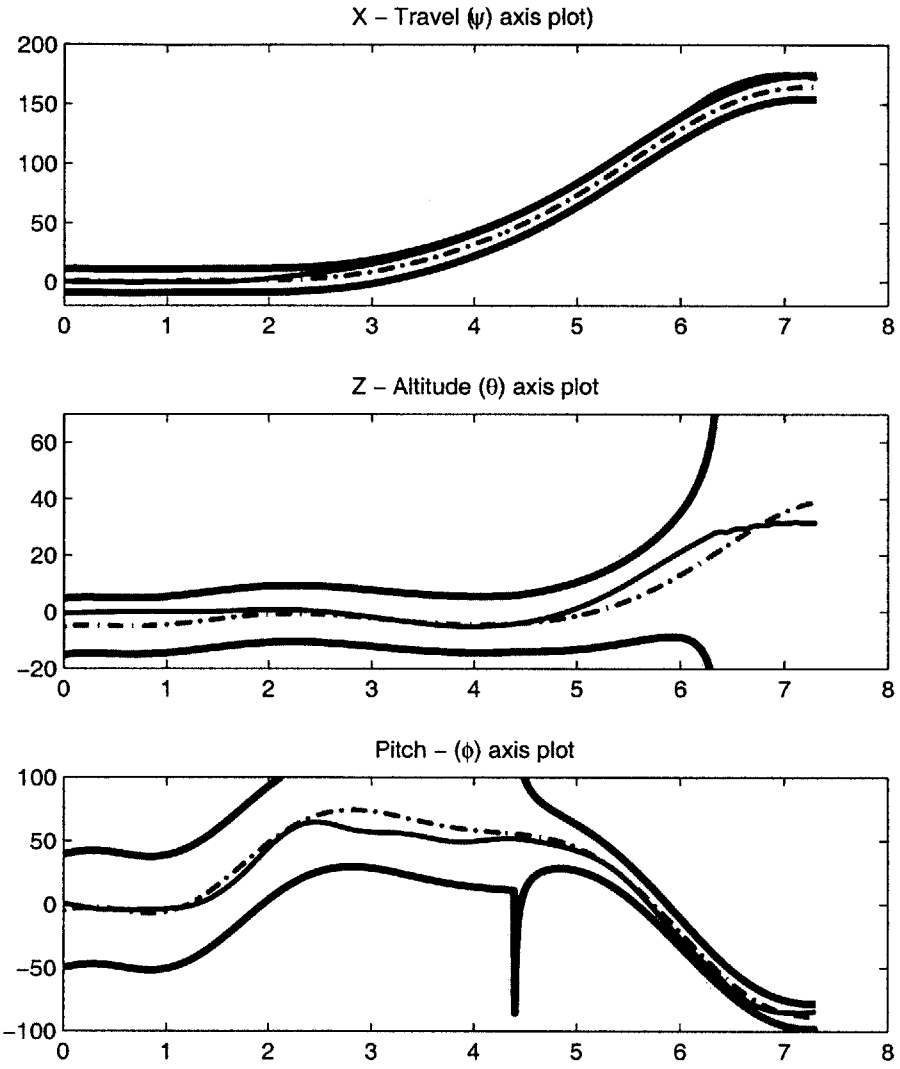


Figure 3-5: Vertical landing experiment, actual and desired trajectory plot.

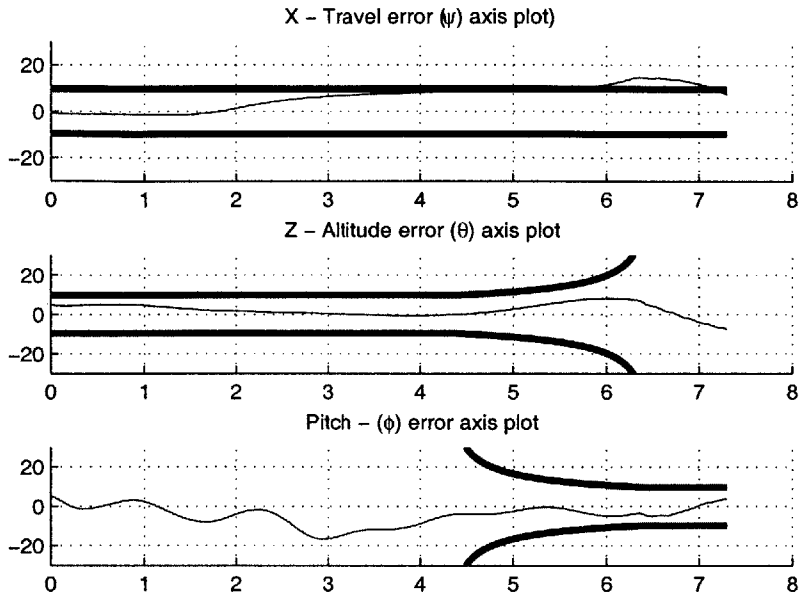


Figure 3-6: Vertical landing experiment, actual and desired trajectory error plot.

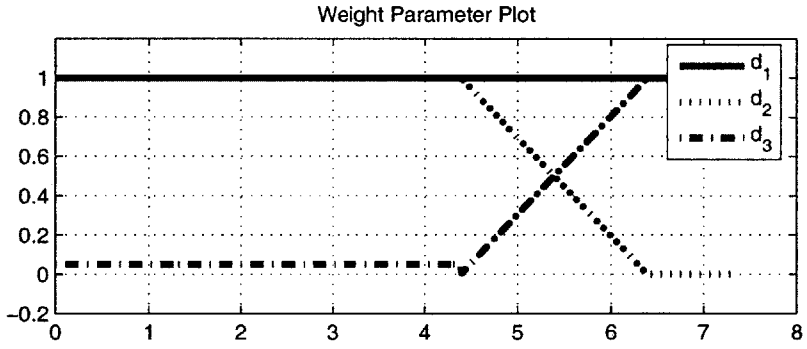
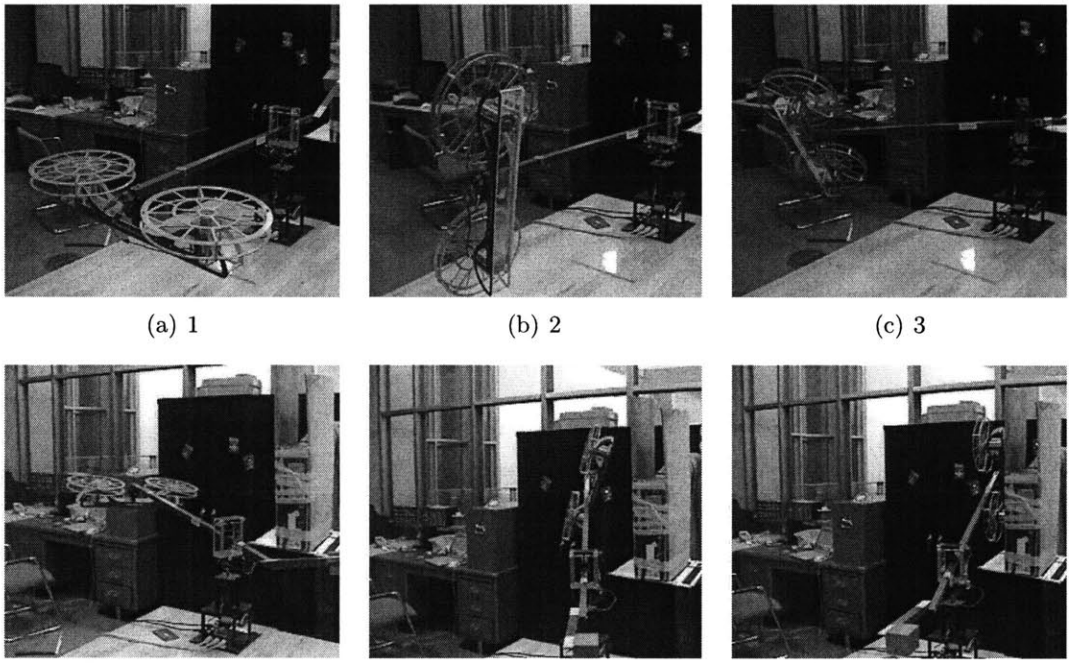


Figure 3-7: Weight parameter plot for the experiment.



(a) 1 (b) 2 (c) 3

(d) 4 (e) 5 (f) 6

Figure 3-8: Snapshots of the vertical landing maneuver.

Chapter 4

Testbed Development

4.1 Overall Testbed Description

In order to demonstrate autonomous aggressive flight capabilities both indoors and outdoors we designed a helicopter UAV platform. As the engineering of the total system is still in the development phase, some of the subsystems like the active stereo positioning have been completed. Details of this system are given in the next section. We have also successfully conducted manual flight experiments with the helicopter. In some of the flight tests, dummy weights equaling the weight of the avionics suite was also added to the airframe.

The airframe of the UAV is the commercially available Eolo Pro model helicopter (80 cm rotor) shown in figure 4-1. The choice of this model was based on its agility and being able to carry our minimum required payload for the autopilot without compromising either of the indoor or outdoor flight capability.

Our autopilot is based on the CloudCap Technology's Piccolo II [3]. Weighing only around 220 grams , Piccolo II is a generic sensor suite for small UAVs and also has built in capability to fly fixed-wing aircraft autonomously and also waypoint navigation capability. However Piccolo II lacks the capability to track aggressive trajectories to perform agile maneuvers. Integrating it with our custom software and additional sensors, we hope to be able to develop an aggressive helicopter autopilot system capable of indoor and outdoor flight. Also due to our modular software

Figure 4-1: Eolo Pro model helicopter

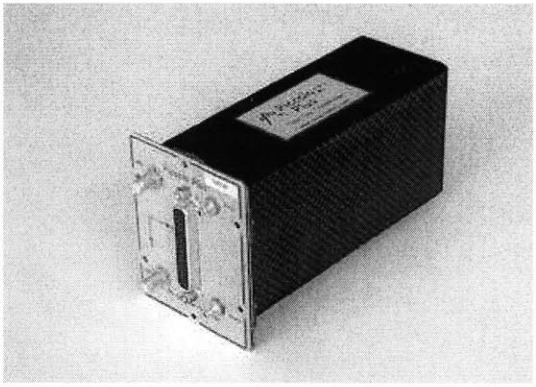


Figure 4-2: Piccolo II avionics box

development effort, only small modifications will be needed to control code in the software for the system to be also used as an agile fixed-wing aircraft autopilot.

Representative block diagrams of the hardware components of the designed helicopter and the ground systems are given in the following figures 4-3 and 4-4 .Dashed blocks represent optional components of the system. As can be understood from the figures, with all the optional blocks removed the system will still have the necessary hardware to perform autonomous functions such as waypoint navigation outdoors only. With the aid of the "active stereo tracking system" the UAV will be localized precisely in close proximity of the ground station (~50 meters) both indoors and outdoors. This will enable precise tracking of aggressive trajectories and also will be used to aid in accurate landing of the UAV. Furthermore a third system which is a

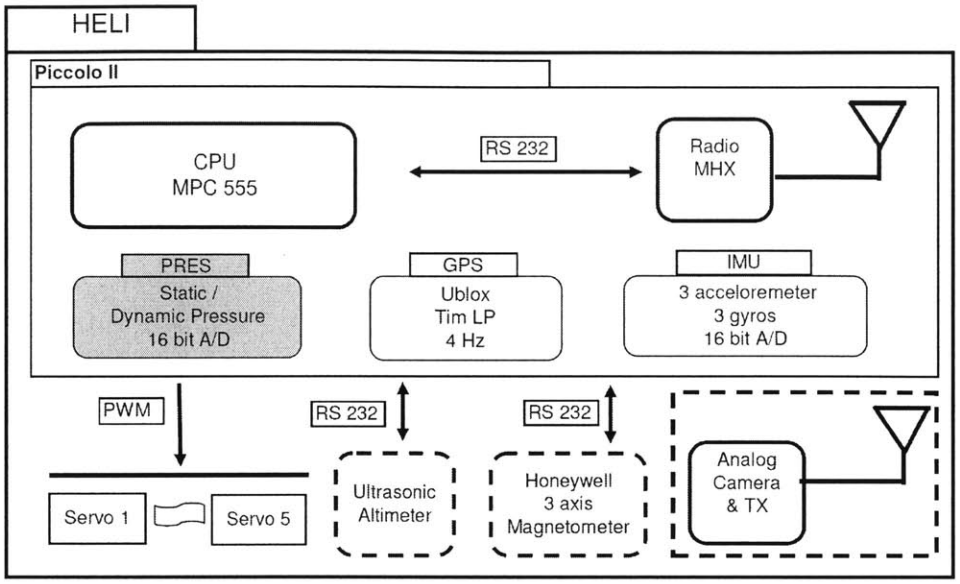


Figure 4-3: Helicopter UAV hardware block diagram

novel relative position and orientation [7] aiding system will be utilized once the UAV is in even closer around 10 meter proximity to the landing pad. A typical application of the two redundant aiding systems would be all weather robust landing on moving platforms. A more exotic application would be "all attitude landings" i.e landing on ceilings and vertical walls. An example scenario would be a ship deck landing. The helicopter is navigates to close proximity of the ship via GPS. Then the active stereo system starts tracking the UAV and brings it closer and finally with the aid of the "moire fringe landing sensor system" the automated landing is performed robustly nearly in all-weather conditions. One might naturally ask the question that why a very primitive and a fundamental function such as landing or takeoff has to rely on so many systems working in harmony. The answer is all of the systems mentioned will be utilized on their availability without failure basis. Meaning they will only be used in improving the solution as long as they supply valuable information to the solution of the problem. Since the information they provide is redundant in nature with different characteristics, this architecture will improve the robustness of the total system.

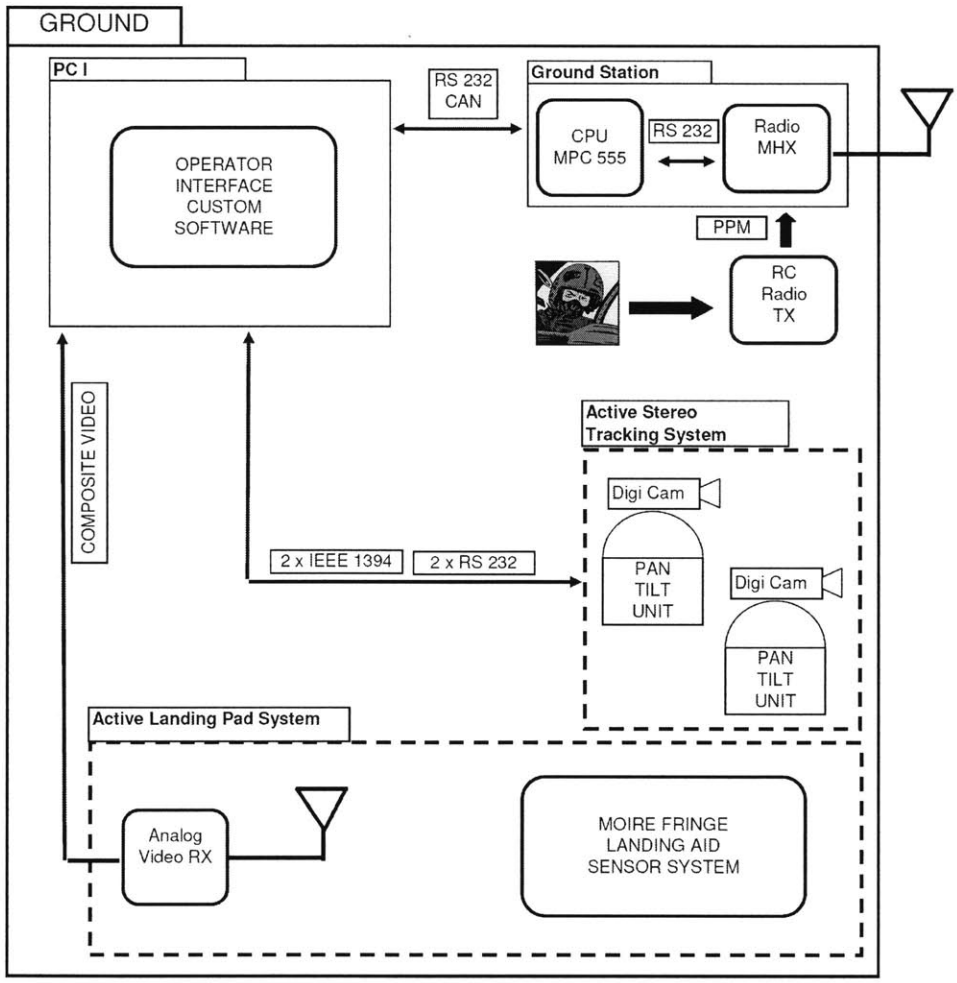


Figure 4-4: Ground station hardware block diagram

4.2 Active Stereo Tracking System

4.2.1 Overall System Design

In our pursuit for a positioning system, we started looking for solutions preferably for COTS (Commercial Off The Shelf). The system had to meet our specific requirements which are:

- Able to work both indoors and outdoors also be able to work together with GPS (Global Positioning System).
- Needs to be have a range of at least 50m. It should be able to localize our helicopter UAV.
- Has the necessary update rate around 20 Hz and positioning accuracy of roughly 20 cm.
- Needs to be able track for our experimental research platform which is a mini helicopter UAV as it performs aggressive maneuvers in the specified range,
- Needs to be relatively cheap.

After searching for products, we concluded that we had to engineer such a system for our application. The active stereo solution with active beacons, seemed to be an attractive choice. Since it matches up with the criteria mentioned above relatively well compared to other options like, "RF tagging", and ultrasonic beacons.

The system consist of two cameras mounted on two pan tilt units and IR beacons (infrared LEDs) to be mounted on the helicopter. A picture of the system is given in 4-5. The cameras are IEEE-1394 DragonFly Cameras from PointGrey [16]. The pan tilt unit is PTU-D46-17 from Directed Perception [4]. The cameras were modified with IR (infrared) bandpass filters in order to ease the detection of the active beacons. For the remaining part of this section, we assume our camera model and the corresponding projection model to be of a pinhole camera's. So all the expressions, equations in the following sections are based on this assumption.

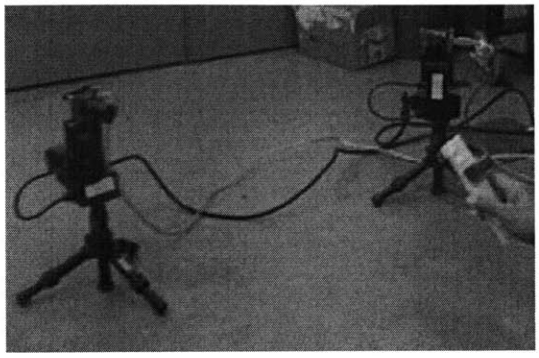


Figure 4-5: Pan-Tilt unit with mounted cameras .

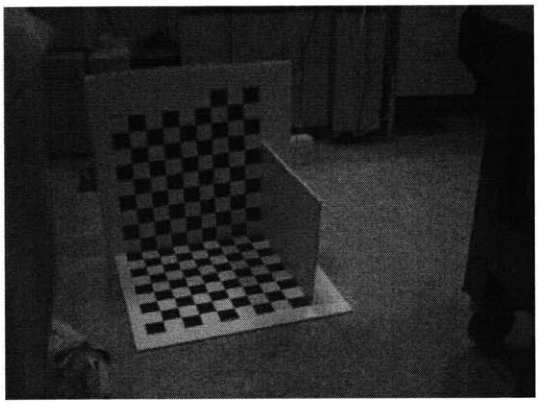


Figure 4-6: Calibration Rig

4.2.2 Camera Calibration

In order to reconstruct 3D position of the target using active stereo camera setup, the camera positions and orientations need to be known with respect to a reference coordinate frame. Camera calibration is the procedure to determine these positions and orientations of the cameras. It is generally achieved using images taken from cameras of a "calibration rig" which has a known position and orientation [8]. There are readily available tools to achieve calibration conveniently. We used the Caltech Camera Toolbox [2] along with a checkerboard pattern calibration rig. One such sample image of the checkerboard taken from our cameras is given in figure 4-6.

We devised the following camera calibration procedure for the pan-tilt unit camera

combination we have. First a local reference coordinate frame is chosen, and the calibration rig is aligned and positioned at the origin of this frame. Then the cameras are placed with a sufficient baseline (distance between camera optical centers) that is determined a priori. This is an important parameter since it determines the disparity in the images and hence affects the uncertainty of the reconstruction. The field of view of the cameras at their initial orientation must contain the calibration rig. Then, several of the rig images are taken with the cameras to extract intrinsic and extrinsic parameters of the cameras. For more detail regarding camera calibration the interested reader can see the reference [8].

4.2.3 Forward Kinematics Map for the Pan-Tilt Unit

Since our cameras are not stationary, the camera calibration matrices change, as the system tracks the target by actively panning and tilting the cameras. We need to compensate for this motion of the cameras to achieve stereo reconstruction of the 3D world point of the target. In other words we need to know position and the orientation of the frames which are attached to the centers of the cameras. We will use the notation used in [15].

We will explain the forward kinematics map associated with the panning and tilting angles of the pan-tilt units with the use of the figure 4-7. Let W denote an absolute world frame aligned with our calibration rig. The origins of both W and the rig are coincident as well. Let $g_{ws} \in SE(3)$ be our initial configuration at the time of our initial calibration and we will also define our panning and tilting angles θ_1, θ_2 respectively to be equal to 0. As our pan-tilt angles change we will define $g_{wt} : \mathbb{S}^1 \times \mathbb{S}^1 \mapsto SE(3)$ to be the current configuration of our camera center with respect to the world frame. $\mathbb{S}^1$ denotes the unit circle since we are dealing with angles. From here, we need to determine, $g_{wt}(\theta_1, \theta_2) \in SE(3)$ *where* $\theta_1, \theta_2 \in \mathbb{S}^1$. We can easily see that:

$$g_{wt}(\theta_1, \theta_2) = g_{ws}\ g_{st}(\theta_1, \theta_2) \tag{4.1}$$

Then we must determine $g_{st}(\theta_1, \theta_2)$. This may be achieved by using the *product of*

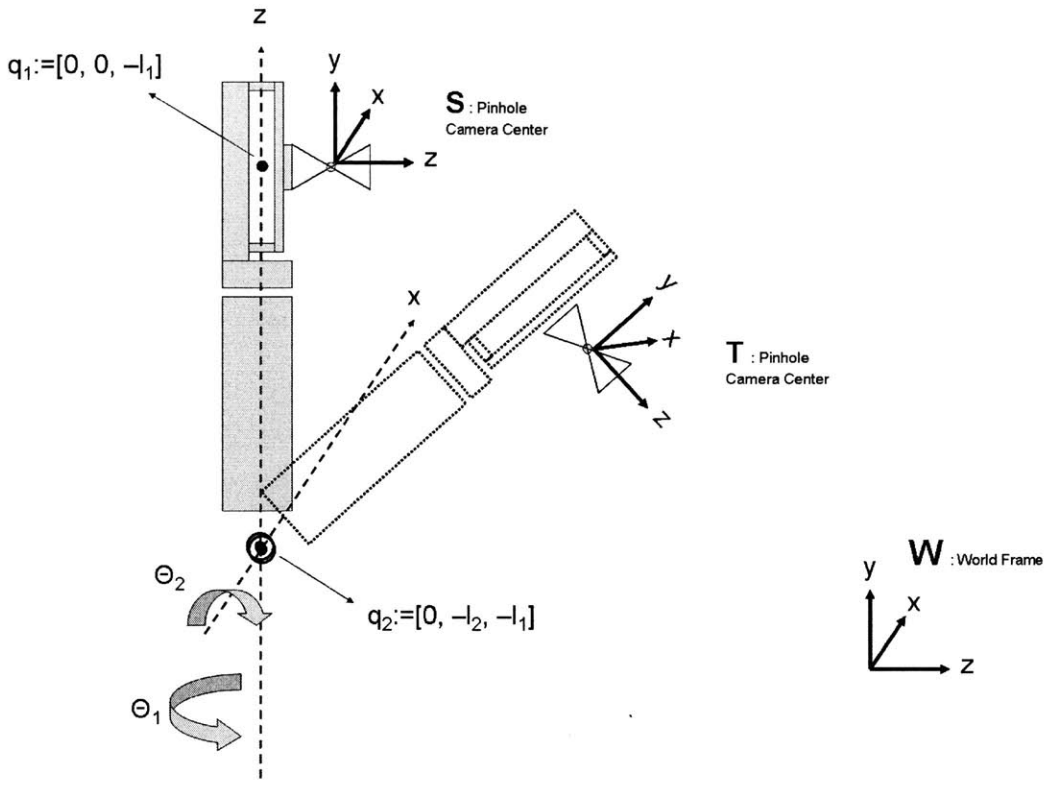


Figure 4-7: Pan-tilt unit coordinate frames and revolute joint diagrams

exponentials formula given in [15]. But we first need to define $\hat{\xi}_1, \hat{\xi}_2 \in se(3)$ as the corresponding twist motions for the revolute joints of the pan-tilt unit represented in frame S. To clarify the notation before proceeding any further, we give the definitions of $\xi, \hat{\xi}$ by the expressions in 4.2 where they represent the vector and the skew-symmetric forms of the same twist, respectively. We can evaluate the expressions for these twist motions as given in 4.4.

$$\xi \equiv \hat{\xi} \; where \; \xi = \begin{bmatrix} v \\ w \end{bmatrix}, \hat{\xi} = \begin{bmatrix} \hat{w} & v \\ 0 & 0 \end{bmatrix} \quad v, w \in \mathbb{R}^3 \; \hat{w} \in so(3) \qquad (4.2)$$

$$q_1 = \begin{bmatrix} 0 \\ 0 \\ -l_1 \end{bmatrix} \quad w_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad q_2 = \begin{bmatrix} -l_2 \\ 0 \\ -l_1 \end{bmatrix} \quad w_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$
$$\xi_1 = \begin{bmatrix} -w_1 \times q_1 \\ w_1 \end{bmatrix} \quad \xi_2 = \begin{bmatrix} -w_2 \times q_2 \\ w_2 \end{bmatrix} \tag{4.3}$$

Now we can proceed to evaluate $g_{st}(\theta_1, \theta_2)$ with the products of exponentials formula given by 4.4.

$$g_{st}(\theta_1, \theta_2) = \exp(\hat{\xi}_1\theta_1)\exp(\hat{\xi}_2\theta_2)g_{st}(0) \quad where \quad g_{st}(0) = I_{4\times4} \tag{4.4}$$

Using equation 4.1 we can easily evaluate $g_{wt}(\theta_1, \theta_2)$ with the aid of a symbolic mathematics package to be as given in 4.5.

$$g_{wt}(\theta_1, \theta_2) = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1)\sin(\theta_2) & \cos(\theta_2)\sin(\theta_1) & d_1 \\ 0 & \cos(\theta_2) & -\sin(\theta_2) & d_2 \\ -\sin(\theta_1) & \cos(\theta_1)\sin(\theta_2) & \cos(\theta_1)\cos(\theta_2) & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$where \ [d_1 \ d_2 \ d_3]^T \ is \ given \ by:$$

$$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} \sin(\theta_1)l_1 + \sin(\theta_1)\left((-1+\cos(\theta_2))\,l_1 + \sin(\theta_2)l_2\right) \\ -\left(\sin(\theta_2)l_1\right) - \left(1-\cos(\theta_2)\right)l_2 \\ \left(-1+\cos(\theta_1)\right)l_1 + \cos(\theta_1)\left((-1+\cos(\theta_2))\,l_1 + \sin(\theta_2)l_2\right) \end{bmatrix} \tag{4.5}$$

4.2.4 Detection

Once the cameras are calibrated, we can determine the 3D position of our target. Our target as mentioned before was an IR beacon (infrared LEDs). With the help of our bandpass infrared filters, under most lighting conditions, if the aperture and

exposure parameters of the cameras are properly setup, the target shows up as the brightest region in the image. So the detection problem breaks down simply into thresholding and the connected region detection problem. Both thresholding region detection can be done extremely efficiently on relatively fast computer. Once the region which represents the target is detected in both images, we use the "center of mass" of the region as the corresponding image point.

4.2.5 3D Localization

Since we have detected the corresponding projection of the target in both images from the two cameras, we can proceed to reconstructing the actual location of the target in the world frame W that we defined before. We use the simple linear triangulation method given in [11]. Let $X_w = [x_w \; y_w \; z_w \; 1]^T \in \mathbb{R}^4$ be the homogeneous coordinate representation of target's position in the world frame W. Furthermore define, $\mathbf{x}_i = [x_i \; y_i \; s_i]^T \in \mathbb{R}^3 \; for \; i = 1, 2$ be the homogeneous coordinate representations of the image coordinates in the cameras 1 and 2 respectively. We have the following set of projection equations 4.6 from both cameras:

$$\begin{aligned} \mathbf{x}_1 &= P_1 X_w \\ \mathbf{x}_2 &= P_2 X_w \end{aligned} \tag{4.6}$$

where $P_i = [K_i \mid O_{3\times1}]\, g_{tw_i}(\theta_1, \theta_2)$, $P_i \in \mathbb{R}^{3\times4}$ $g_{tw_i} = g_{wt_i}^T \; for \; i = 1, 2$. And $K_i \in \mathbb{R}^{3\times3}$ are the intrinsic camera calibration matrices as given in [8]. We have six equations and three unknowns, though two of the equations provide no information since they are scale factors. Following the same procedure as in [11] to eliminate the scale factors, we operate on both sides of the equations with the cross product $\mathbf{x}_i \times \mathbf{x}_i = \mathbf{x}_i \times (P_i X_w)$ to get the equivalent set of homogeneous equations $\mathbf{x}_i \times (P_i X_w) = 0$. Eliminating the linearly dependent equation from both sets we end up with four equations and three unknowns. Writing these equations in matrix form as $AX_w = b$ where $A \in \mathbb{R}^{4\times3}$, $b \in \mathbb{R}^4$. The explicit form of A, *and* b are given in 4.7.

$$
\begin{aligned}
A &= \begin{bmatrix} (x_1 p^1_{31} - p^1_{11}) & (x_1 p^1_{32} - p^1_{12}) & (x_1 p^1_{33} - p^1_{13}) \\ (y_1 p^1_{31} - p^1_{21}) & (y_1 p^1_{32} - p^1_{22}) & (y_1 p^1_{33} - p^1_{23}) \\ (x_2 p^2_{31} - p^2_{11}) & (x_2 p^2_{32} - p^2_{12}) & (x_2 p^2_{33} - p^2_{13}) \\ (y_2 p^2_{31} - p^2_{21}) & (y_2 p^2_{32} - p^2_{22}) & (y_2 p^2_{33} - p^2_{23}) \end{bmatrix} \\
b &= \begin{bmatrix} p^1_{14} - x_1 p^1_{34} \\ p^1_{24} - y_1 p^1_{34} \\ p^2_{14} - x_2 p^2_{34} \\ p^2_{24} - y_2 p^2_{34} \end{bmatrix} \\
&\text{where } p^k_{ij} \text{ are the corresponding elements of } P_k
\end{aligned} \tag{4.7}
$$

We are left with overdetermined sets of equations,and then we proceed to take the minimum least squares given by 4.8.

$$X_w = (A^T A)^{-1} A^T b \tag{4.8}$$

Please note that only under perfect measurements will the vector b be in the column space of the matrix A. In other words, the rays that are back-projected from the corresponding image plane points (through the respective camera centers), will only intersect in the three dimensional world frame when we have noiseless measurements. For this to happen is practically almost impossible, since even due finite camera resolution will the measurements be noisy. Thus, we are forced to such a "error norm" minimizing solution.

4.2.6 Error Covariance Calculation

As mentioned in the prior section, the measurements will be noisy due many sources error. Some of these are, errors due to imperfect calibration, which can arise from mechanical precision tolerances, detection errors in the image plane, quantization errors, etc. Nevertheless, since the world position estimates will be fed into an EKF (Extended Kalman Filter) estimator, we would like to determine the uncertainties of

our position estimates. A reasonable model for our measurement noise would be the white Gaussian noise assumption. We assume that our image plane measurements given by $\hat{\mathbf{x}}_i = \mathbf{x}_i + \mathbf{v}_i$ are corrupted with $\mathbf{v}_i \sim N(0, \Lambda_i)$ where $\Lambda_i \in \mathbb{R}^{2\times 2}$ are the associated covariance matrices for the noise in the images. Since the mapping $X_w = f(\mathbf{x})_1, \mathbf{x}_2)$ given in 4.8 is nonlinear. The probability distribution of the error of $f(\cdot,\cdot)$ will not necessarily be Gaussian. This would violate the requirement of our estimator and also the math involved would be too complicated to be practical. Instead, we use the Taylor expansion of $f(\cdot,\cdot)$ given in 4.9 up to the first order as suggested by [6]:

$$f(\mathbf{x}_1 + \mathbf{v}_1, \mathbf{x}_2 + \mathbf{v}_2) \approx f(\mathbf{x}_1, \mathbf{x}_2) + J(\mathbf{x}_1, \mathbf{x}_2) \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} \tag{4.9}$$

where $J(\mathbf{x}_1, \mathbf{x}_2) = (\nabla_{\mathbf{x}_1,\mathbf{x}_2} f)^T(\mathbf{x}_1, \mathbf{x}_2)$. So the covariance of V_X in $\hat{X}_w = X_w + V_X$ could be approximated as given below in 4.10.

$$\Lambda_W = J(\mathbf{x}_1, \mathbf{x}_2) \underbrace{\begin{bmatrix} \Lambda_1 & O_{2\times 2} \\ O_{2\times 2} & \Lambda_2 \end{bmatrix}}_{\Lambda_v} J^T(\mathbf{x}_1, \mathbf{x}_2) \tag{4.10}$$

which follows from:

$$V_X \approx J(\mathbf{x}_1, \mathbf{x}_2)\mathbf{v}_c \quad \textit{where } \mathbf{v}_c = [\mathbf{v}_1 \ \mathbf{v}_1]^T \tag{4.11}$$

$$E\left[V_X V_X^T\right] \approx E\left[J\mathbf{v}_c\mathbf{v}_c^T J^T\right] = JE\left[\mathbf{v}_c\mathbf{v}_c^T\right] J^T = J\,\Lambda_v\,J^T \tag{4.12}$$

Chapter 5

Conclusions and Future Directions

In this thesis, we have presented a novel approach to nonlinear vehicle trajectory tracking, when the important vehicle outputs change as a function of its state relative to some external environment. Considering the scenario of aggressive vehicle landing in demanding conditions, we have illustrated our method, which consists of generating control inputs that properly "interpolate" the system output of interest, with initial emphasis on position, but later emphasis on attitude as the vehicle comes close to landing. While the theoretical foundation of our work remains to be established, several simulations and laboratory experiments already indicate the potential usefulness of our approach. Natural extensions to this approach would be the:

- Incorporation of the full vehicle dynamics, including the lateral dynamics along with the longitudinal dynamics.
- A systematic analysis and inclusion of actuator constraints in the given framework could potentially be very useful for practical implementations.
- Much also remains to be done in terms of both the modeling environment and obstacles, either of which could be static or dynamic.
- Exploitation of the symmetries and invariants suggested by [9], in the state space with regards to the dynamics, could potentially generalize the use of many stored maneuvers to land at different locations and configurations.

Appendix A

Extended Kalman Filter

A.1 Filter Description

There has been a wealth of literature published over the last decades regarding the use of Kalman filters for the purposes of GPS aided inertial navigation. In this section, we will explain the details of our implementation of an Extended Kalman Filter, which will be used to estimate the states necessary for our controllers. We will explain the details of our implementation without going too much into the theory as it is well known and well documented in the literature. Interested reader may look at the references listed [1].

Our model implementation is based on real sensors available on a typical UAV avionics sensor suite (Piccolo II by CloudCap Tech.) [3]. Specifically, our sensors consist of three strapdown gyros, three accelerometers in conventional orthogonal placement and a GPS sensor. The gyros and the accelerometers were sampled at a rate of 20 Hz, the GPS gives positional and velocity information at 4 Hz. Data from the IMU (inertial sensors, gyros, accelerometers) were collected in the lab for characterizing the spectral properties of the noise of the sensors. GPS noise model was assumed to be Gaussian white with typical covariance levels documented from the literature and based on prior experience with this sensor. Please note that all noise sources from the sensors were assumed to be white Gaussian as this is a necessary assumption for filter optimality.

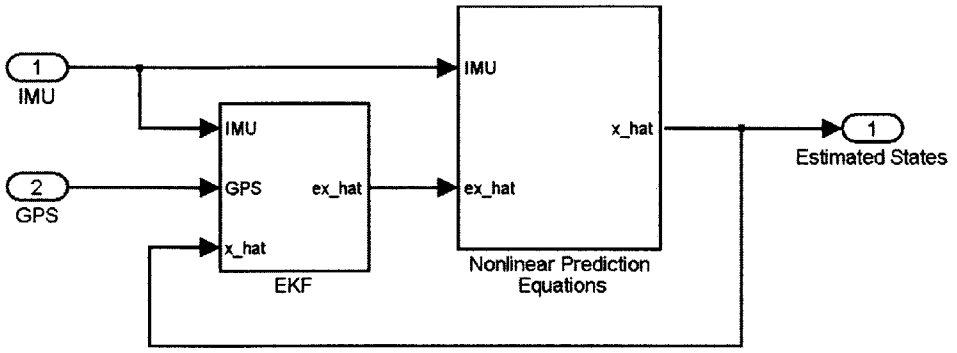


Figure A-1: EKF Filter block diagram "tightly coupled" EKF filter

A schematic of the filter implementation is given in the figure A-1. This is a typical example of a GPS aided INS (Inertial Navigation System) implementation with an Extended Kalman Filter. This specific block diagram resembles the "tightly coupled EKF" as it is referred to in the literature. [20].

A.1.1 Dynamic Model

Our model dynamics are based on kinematic equations of motion as given in [10]. This property makes the filter independent of the specifics of the vehicle dynamics. Thus the filter could be as used as well with any other system having similar sensors i.e helicopters, fixed-wing aircraft, land based vehicles, etc. For attitude representation quaternions were used due to significant advantages they have in terms of being singularity free and numerical error behaviour [5]. Our equations of motion are given below in A.1.

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= R_{tb}\,(a_m - a_b) - g + R_{tb}\,u_a \\ \dot{q} &= (\hat{\Psi}_m - \hat{\Psi}_b)\,q + \hat{u}_g\,q \\ \dot{a}_b &= 0 \\ \dot{\psi}_b &= 0 \end{aligned} \tag{A.1}$$

The system has sixteen states and they are as follows.

States and Relevant Terms

- x: Position in the local inertial NED (North East Down) frame. $[x\ y\ z]$
- v: Inertial velocity in the NED frame $[v_x\ v_y\ v_z]$
- q: Quaternion representation of attitude $[q_0\ q_1\ q_2\ q_3]$
- a_b: Accelerometer biases on 3 accelerometers $[a_{bx}\ a_{by}\ a_{bz}]$
- Ψ_b: Gyros biases on 3 gyros $[\psi_{bx}\ \psi_{by}\ \psi_{bz}]$

Note: hat over Ψ_b, Ψ_m and u_g as given in A.1 denotes in 4x4 skew symmetric matrix form used for quaternions as given below A.2.

$$\hat{\Psi} = \frac{1}{2}\begin{bmatrix} 0 & \psi_z & -\psi_y & \psi_x \\ -\psi_z & 0 & \psi_x & \psi_y \\ \psi_y & -\psi_x & 0 & \psi_z \\ -\psi_x & -\psi_y & -\psi_z & 0 \end{bmatrix} \tag{A.2}$$

- R_{tb}: 3x3 Rotation matrix element of the group SO(3) . It is a function of the quaternions in this case and is used to transform body accelerations in to NED inertial frame. The matrix in terms of quaternion attitude representation is given below A.3.

$$R_{tb} = \begin{bmatrix} q_0^2+q_3^2-q_1^2-q_2^2 & 2(q_0q_1-q_2q_3) & 2(q_0q_2+q_1q_3) \\ 2(q_0q_1+q_2q_3) & q_1^2+q_3^2-q_0^2-q_2^2 & 2(q_1q_2-q_0q_3) \\ 2(q_0q_2-q_1q_3) & 2(q_1q_2+q_0q_3) & q_2^2+q_3^2-q_0^2-q_1^2 \end{bmatrix} \tag{A.3}$$

Inputs

- a_m: Measured acceleration vector in vehicle body frame $[a_{mx}\ a_{my}\ a_{mz}]$
- Ψ_m: Measured angular rate vector in vehicle body frame $[\psi_{mx}\psi_{my}\ \psi_{mz}]$
- g: Local gravity vector assumed to be $[0\ 0\ g]$

- u_a: Accelerometer noise input vector in vehicle body frame (Gaussian white) $[u_{ax}\ u_{ay}\ u_{az}]$

- u_g: Gyro noise input vector (Gaussian white) $[u_{gx}\ u_{gy}\ u_{gz}]$

Outputs

- z_x: GPS position in the local inertial NED (North East Down) frame. $[x\ y\ z]$

- z_v: GPS velocity in the NED frame $[v_x\ v_y\ v_z]$

- u_x: Gaussian white GPS position noise

- u_v: Gaussian white GPS velocity noise

A.1.2 Linearization around Estimated state

We need to linearize our nonlinear equations around the estimated state in order to propagate Gaussian process noise covariance with the 1st order approximation of the nonlinear equations of motion. Linearized continuous time state propagation equations will be of the form A.4.

$$\dot{x} = Fx + Bu + Gn \tag{A.4}$$

where $E\left[n(t)n^T(\tau)\right] = W\delta(t-\tau)$. Note x denotes the full state as given by our model (16x1). u denotes our inputs and n denotes white Gaussian noise inputs.

Our discrete time measurement equations will be of the form A.5

$$z[k] = H\,x[k] + v[k] \tag{A.5}$$

where $E[v[i]\,v^T[j]] = R\ when\ i = j$ and $E[v[i]\,v^T(j)] = 0\ when\ i \neq j$.

We will continue by explicitly constructing the terms of the equation A.4.

The F Jacobian matrix will be as A.6:

$$F(x, a_m, \Psi_m) = \begin{bmatrix} O_{3\times3} & I_{3\times3} & O_{3\times4} & O_{3\times3} & O_{3\times3} \\ O_{3\times3} & O_{3\times3} & \partial_{3\times4} & R_{3\times3} & O_{3\times3} \\ O_{4\times3} & O_{4\times3} & \hat{\Psi}_{4\times4} & O_{4\times3} & Q_{4\times3} \\ O_{6\times3} & O_{6\times3} & O_{6\times4} & O_{6\times3} & O_{6\times3} \end{bmatrix} \tag{A.6}$$

The $I_{i\times j}$ and $O_{i\times j}$ denote identity and zero matrices of the corresponding size.

The sub block of the matrix marked with the $\partial_{3\times4}$ symbol is:

$$\frac{\partial}{\partial q} R_{tb}(a_m - a_b)$$

which when evaluated symbolically is too big to fit in here. So it is not given. The code is included in Appendix B.

The sub block marked with R is:

$$R_{tb}$$

The sub block given by $\hat{\Psi}$ (in skew symmetric form) is :

$$\hat{\Psi}_m - \hat{\Psi}_b$$

Lastly the sub block marked with $Q_{4\times3}$ is:

$$Q_{4\times3} = -\frac{1}{2} \begin{bmatrix} q_3 & -q_2 & q_1 \\ q_2 & q_3 & -q_0 \\ -q_1 & q_0 & q_3 \\ -q_0 & -q_1 & -q_2 \end{bmatrix}$$

The B matrix and the G matrix of A.4 are the same and they are given by:

$$B = G = \begin{bmatrix} O_{3\times3} & O_{3\times3} \\ R_{tb} & O_{3\times3} \\ O_{4\times3} & Q_{4\times3} \\ O_{6\times3} & O_{6\times3} \end{bmatrix}$$

Furthermore we need to discretize the linearized continuous state propagation equations to put them into the form necessary for digital computer implementation. Our IMU measurements were sampled at a rate 20 Hz. We assume that they are approximately constant between samples.

The linearized discrete state propagation equations will be of the form given in A.7.

$$x[k+1] = \Phi\, x[k] + \Gamma\, u[k] + \Omega\, n[k] \tag{A.7}$$

where $E[n(i)n^T(j)] = W$ *when* $i = j$ and $E[n(i)n^T(j)] = 0$ *when* $i \neq j$. Along with measurement equations above one more assumption we need to make here is state noise $n[k]$ is not correlated with $u[k]$. Formally stating $E[n[i]v^T[j]] = 0$ *for all* i, j.

The expressions for evaluating the Φ and the Q matrices are as given by the [1]. They are repeated here in A.9 for clarity.

$$A = \begin{bmatrix} -F & GWG^T \\ 0 & F^T \end{bmatrix} \Delta t \tag{A.8}$$

$$M = \exp A = \begin{bmatrix} \bullet & \Phi^{-1}Q \\ 0 & \Phi^T \end{bmatrix} \tag{A.9}$$

Using the two subblocks of the matrix M we can easily evaluate the state transition matrix Φ and the process noise covariance matrix Q. The Γ and the Ω matrices will be omitted since they are not necessary for the implementation. Measurement update covariance propagation equations are also omitted since they are standard. Interested readers can refer to [1].

A.1.3 Prediction Step: Solutions of the Nonlinear Equations of Motion

Though generally it not possible to obtain analytical solutions for nonlinear odes (Ordinary Differential Equations), it was possible in this case.In the implementation the nonlinear equations were solved numerically. They are not given here interested reader may contact the author.

A.1.4 Sensor Noise Models

The IMU (inertial measurement unit) of the Piccolo II was used to collect data from the 3 gyros and 3 accelerometers. As noted earlier, the Gaussian white noise assumption had to be made on the sensor characteristics. The IMU sensor noises were also assumed to be statistically independent of each other. The resulting covariance matrices based on 5000 sample data points for accelerometers and gyros were evaluated to be:

$$E[u_a u_a^T] = diag\,[9.05 \times 10^{-05}\ ,\ 1.96 \times 10^{-04}\ ,\ 1.43 \times 10^{-04}]$$

$$E[u_\Psi u_\Psi^T] = diag\,[2.12^{-05}\ ,\ 8.72 \times 10^{-06}\ ,\ 1.51 \times 10^{-05}]$$

GPS measurement noise were again assumed to be a white noise sequence with Gaussian distribution and based on crude approximations which are based on previous experience with this sensor before. They are given below .

$$E[u_x u_x^T] = diag\,[5\ ,\ 5\ ,\ 25]$$

$$E[u_v u_v^T] = diag\,[0.5\ ,\ 0.5\ ,\ 1]$$

In addition a sample plot of the typical sensor noise is given in figures A-2 and A-3 based on collected data.

An interesting point worth noting here is that without an absolute heading reference during steady and level flight with constant velocity, the heading becomes

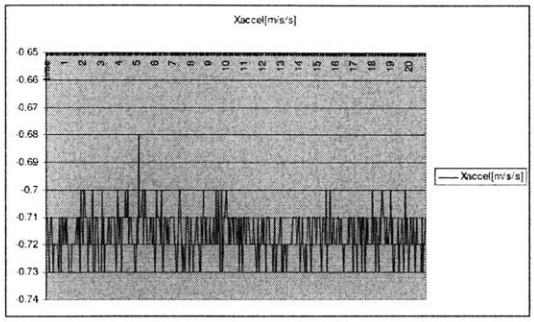


Figure A-2: Accelerometer sensor sample plot

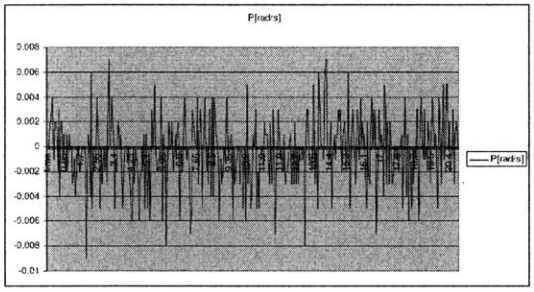


Figure A-3: Gyro sensor sample plot

unobservable with the combination of sensors used [4] . A detailed analysis of the observability of the certain modes of the system along various trajectories is omitted here for simplicity sake.

A.1.5 Simulink Implementation

Despite the wealth of literature on the subject matter, implementation of the filter turned out to be a very painful task due to complicated symbolic expressions that needed to be evaluated and also due to the relatively large dimensions of the filter, namely sixteen. Simulink codes of the overall EKF implementation are given in the next section of the appendix. Implementation of the filter is mainly done in the custom written in an embedded m-function block.

A.1.6 Analysis of the EKF Performance

As noted before getting the filter to work properly was a very painful task . Also given the fact that EKF is an ad hoc procedure with no theoretical guarantees of converges of the state error estimates, a lot of the sources of implementation mistakes were hard to identify.

The EKF does a reasonable job with respect to keeping the state estimate errors low. Please note that as predicted for steady level flight the heading error diverges over long time horizons as it was expected. So some maneuvering in the bank was added to achieve convergent behavior. Error plots of estimated states for sample run of 50 seconds are given in figures A-4 and A-5.

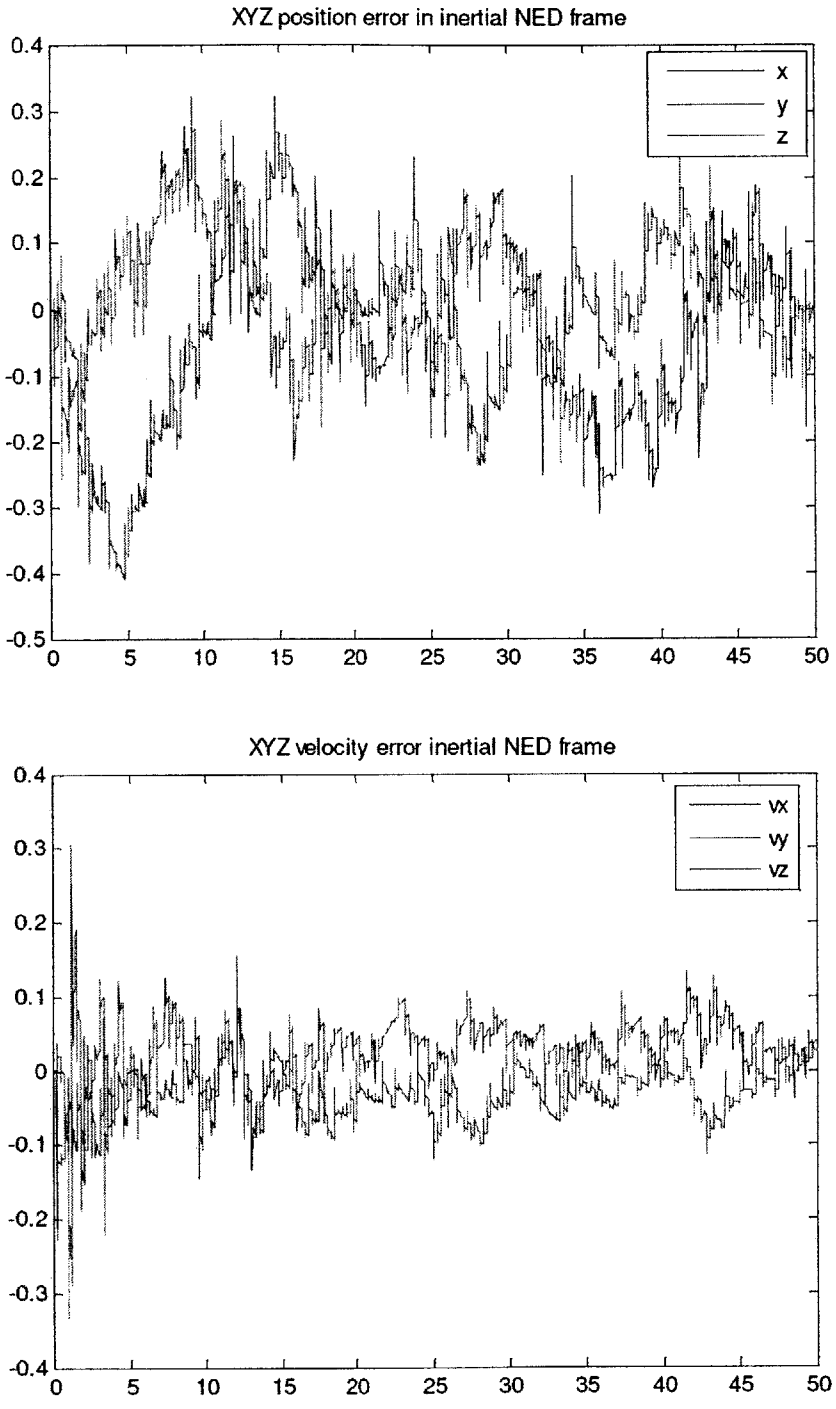


Figure A-4: XYZ position and velocity state estimation error plots

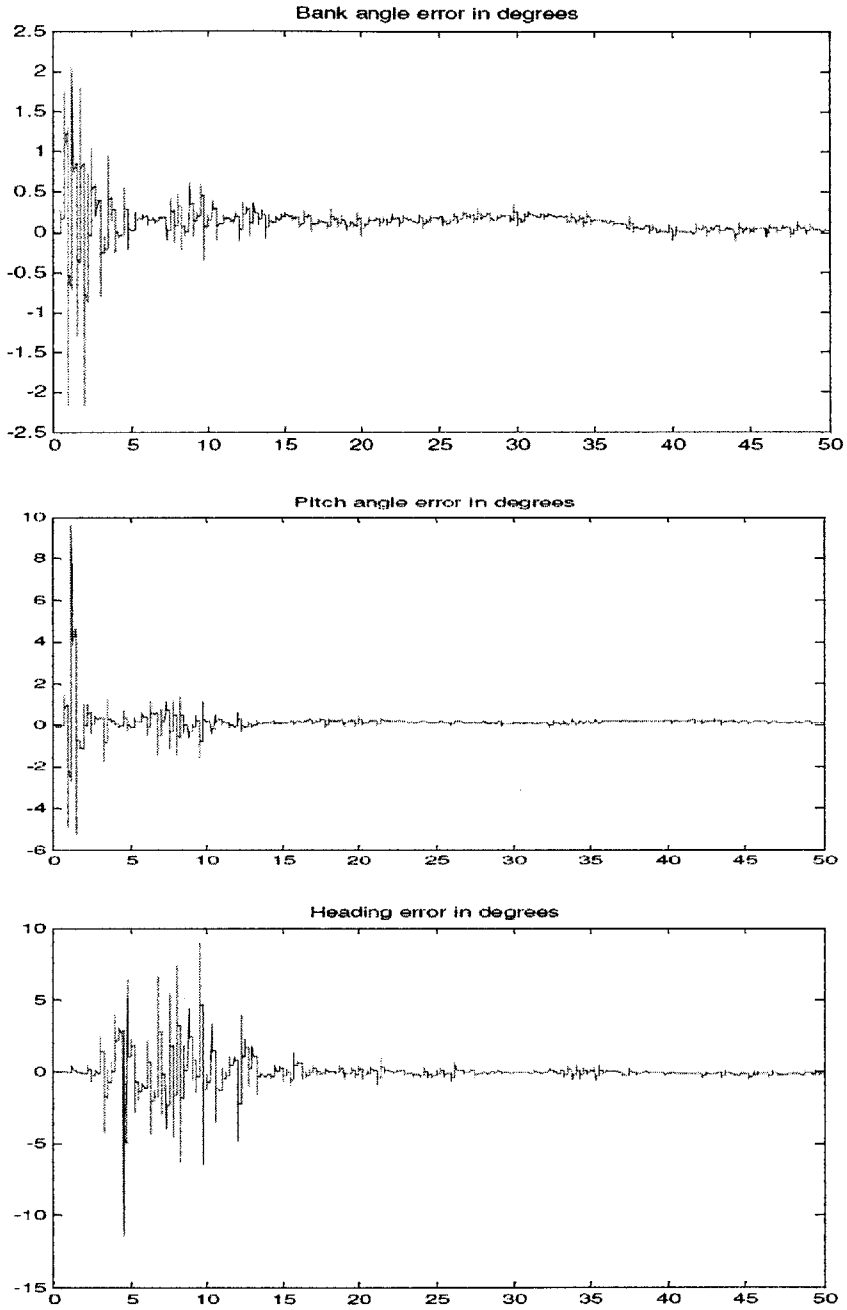


Figure A-5: Attitude state estimation error plots

Appendix B

EKF Codes

```
function [Pout,ex,DCM]  = EKF3D(PSIm,Am,x,z,t)
% function [Pout,ex]  = EKF3D(PSIm,Am,x,z,t)
% Selcuk Bayraktar  Dec 2005 - MIT
% EKF 3D update equations
% inputs are as follows
% PSIm: measured body rotations rates in body frame rad/s [p q r]
% Am: measured body accelerations in body frame m/s^2 [ax ay az]
% x: State vector [px py pz vx vy vz q0 q1 q2 q3 abx aby abz psibx
 psiby psibz]
%   px-z are positions vx-z are velocities
%   q0 - q4 are quaternions for attitude rep.
%   abx-z are accelerometer biases psix-z are gyro biases
% z: GPS measurement update [x y z vx vy vz]
% t: is simulation time
% ------------------------------------------------------------
% Outputs are as follows:
% Pout: 16x16 State Covariance
% ex: Estimated State error

% Big Time this was a lot of trouble
```

```
% This block supports an embeddable subset of the MATLAB language.

persistent F;
persistent P_minus;
persistent Q;
persistent R;
persistent ex_hat;
persistent H;
persistent time_t;
persistent Rtb;
persistent W;

if (isempty(H))
H=[eye(6) zeros(6,10)];
end

if (isempty(F))
    F=zeros(16);
end

if (isempty(P_minus))
    P_minus=eye(16)*0.001;
    P_minus(1:6,1:6)=1*eye(6);
    P_minus(11:16,11:16)=0.1*eye(6);
end

if (isempty(Q))
    Q=eye(16)*1e-6;
    Q(11:13,11:13)=2e-4*eye(3);
    Q(14:16,14:16)=2e-5*eye(3);
```

```
    W=Q(11:16,11:16);
end

if (isempty(R))
    R=0.5*eye(6);
    R(1:3,1:3)=diag([5 5 25]);
end

if (isempty(ex_hat))
    ex_hat=zeros(16,1);
end

if (isempty(time_t))
    time_t=t;
end

if (isempty(Rtb))
    Rtb=eye(3);
end

%%%DoImuUpdate  T=0.05s (20Hz)
if((mod(round(t*1000),0.05*1000)==0)&&((time_t~=t)))
%% init states here as well
%% and everything else that needs to be
    P=zeros(16);
% Let me pull some of the states necessary from the state vector x
    q0=x(7);
    q1=x(8);
    q2=x(9);
    q3=x(10);
```

```
    abx=ex_hat(11);
    aby=ex_hat(12);
    abz=ex_hat(13);
    Pb=ex_hat(14);
    Qb=ex_hat(15);
    Rb=ex_hat(16);

% Let me pull the IMU measurements
    Pm=PSIm(1);
    Qm=PSIm(2);
    Rm=PSIm(3);
    amx=Am(1);
    amy=Am(2);
    amz=Am(3);

%% Jacobian Matrix
    Rtb=[(q0^2)-(q1^2)-(q2^2)+(q3^2) 2*(q0*q1+q2*q3) 2*(q0*q2-q1*q3);...
    2*(q0*q1-q2*q3) -(q0^2)+(q1^2)-(q2^2)+(q3^2) 2*(q1*q2+q0*q3);...
    2*(q0*q2+q1*q3) 2*(q1*q2-q0*q3) -(q0^2)-(q1^2)+(q2^2)+(q3^2)];
    Rtb=Rtb';

    delq0Rt=[(2*(-abx + amx)*q0+2*(-aby + amy)*q1 + 2*(-abz + amz)*q2);...
            (-2*(-aby + amy)*q0 + 2*(-abx + amx)*q1 - 2*(-abz + amz)*q3);...
            (-2*(-abz + amz)*q0 + 2*(-abx + amx)*q2 + 2*(-aby + amy)*q3)];

    delq1Rt=[(2*(-aby + amy)*q0 - 2*(-abx + amx)*q1 + 2*(-abz + amz)*q3);...
             (2*(-abx + amx)*q0 + 2*(-aby + amy)*q1 + 2*(-abz + amz)*q2);...
             (-2*(-abz + amz)*q1 + 2*(-aby + amy)*q2 - 2*(-abx + amx)*q3)];
```

```
delq2Rt=[(2*(-abz + amz)*q0 - 2*(-abx + amx)*q2 - 2*(-aby + amy)*q3);...
         (2*(-abz + amz)*q1 - 2*(-aby + amy)*q2 + 2*(-abx + amx)*q3);...
         (2*(-abx + amx)*q0 + 2*(-aby + amy)*q1 + 2*(-abz + amz)*q2)];

delq3Rt=[(2*(-abz + amz)*q1 - 2*(-aby + amy)*q2 + 2*(-abx + amx)*q3);...
         (-2*(-abz + amz)*q0 + 2*(-abx + amx)*q2 + 2*(-aby + amy)*q3);...
         (2*(-aby + amy)*q0 - 2*(-abx + amx)*q1 + 2*(-abz + amz)*q3)];

SkewPsiM=[0     Rm/2 -Qm/2  Pm/2;...
        -Rm/2   0    Pm/2  Qm/2;...
         Qm/2  -Pm/2   0    Rm/2;...
        -Pm/2 -Qm/2  -Rm/2   0];

SkewPsiB=[0     Rb/2 -Qb/2  Pb/2;...
         -Rb/2   0    Pb/2  Qb/2;...
         Qb/2  -Pb/2   0    Rb/2;...
         -Pb/2 -Qb/2  -Rb/2   0];
PsiToQuat= 0.5*[q3  -q2  q1;...
                q2   q3  -q0;...
                -q1  q0  q3;...
                -q0 -q1 -q2];

%% Let us Start Forming the Jacobian Matrix
F=zeros(16);
F(1:3,4:6)=eye(3);
F(4:6,7:10)=[delq0Rt delq1Rt delq2Rt delq3Rt];
F(4:6,11:13)=[-Rtb];
F(7:10,7:10)=[SkewPsiM-SkewPsiB];
F(7:10,14:16)=[-PsiToQuat];
```

```
%Let us Form the Transition Matrix Now
tao=0.05;    %% Sampling interval for IMU
PHI=eye(16);
PHI=expm(F*tao);
G=[zeros(3,6);...
   Rtb zeros(3);...
   zeros(4,3) PsiToQuat;...
   zeros(6,6)];
A=[-F G*W*G';...
   zeros(16) F'];
B=zeros(32);
temp=zeros(16);
B=expm(A*tao);
temp=B(1:16,17:32);
Q=PHI*temp;
%% Add some very small noise for the bias terms
%% So that the P covariance matrix does become singular
Q(11:16,11:16)=1e-9*eye(6);

%%  Update Covariance here
if(mod(round(t*1000),0.25*1000)==0)
  %  (DoMeasurementUpdate)%% Do GPS measurent update at T=0.25s (4Hz)

    %% Let us Calculate Calculate optimal K
    K=P_minus*(H')*(inv(H*P_minus*(H')+R));
    I=eye(16);

    %% Covariance update due to measurement
    P_minus=(I-K*H)*P_minus*((I-K*H)')+K*R*K';
```

```
        %%Let us do the update on the state
        %% x_hat= x_hat_minus + K (z-z_hat)
        x_hat_minus=x;
        z_hat=H*x_hat_minus;
        ex_try=K*(z-z_hat);
        ex_hat(1:6)=ex_hat(1:6)+ex_try(1:6);
        ex_hat(7:16)=ex_try(7:16);

        %% The following is only one of the many thousand things I tried.
        %q_new=zeros(4,1);
        %q_norm=norm([ex_hat(7:10)+x(7:10)]);
        %q_new=[ex_hat(7:10)+x(7:10)]/q_norm;
        %ex_hat(7:10)=q_new-x(7:10);
    end  % GPS measurement update

    %% Covariance update for the prediction step
    P_minus=PHI*P_minus*(PHI')+Q;
    %% In the end Assign the new covariance matrix to the old one to keep
    P_minus=P_minus;
    %%

end

time_t=t;
%% Now proceed with outputs
ex=ex_hat;
Pout=trace(P_minus);
DCM=Rtb';
```

Bibliography

[1] Robert Grover Brown and Patrick Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering.* John-Wiley Sons, 1997.

[2] *Caltech Camera Toolbox Website.* www.vision.caltech.edu, last accessed Feb 2006.

[3] *CloudCap Technology Website.* www.cloudcaptech.com, last accessed Feb 2006.

[4] *Directed Perception Website.* www.dperception.com, last accessed Feb 2006.

[5] J. Farrel and M. Barth. *The Global Positioning System and Inertial Navigation.* McGraw-Hill, 1997.

[6] Olivier Faugeras. *Three-Dimensional Computer Vision.* The MIT Press, 1993.

[7] Eric Feron and Jim Paduano. A passive sensor for position and attitude estimation using an interferometric target. In *Proc. 43rd IEEE Conference on Decision and Control*, pages 1663–1669, Bahamas, December 2004.

[8] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach.* Prentice Hall, 2002.

[9] Emilio Frazzoli. *Robust hybrid control for autonomous vehicle motion planning.* PhD thesis, Massachusetts Institute of Technology, Massachusetts, 2001.

[10] Vladislav Gavrilets. *Autonomous aerobatic maneuvering of miniature helicopters.* PhD thesis, Massachusetts Institute of Technology, Massachusetts, August 2003.

[11] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2004.

[12] John Hauser, Shankar Sastry, and George Meyer. Nonlinear control design for slightly non-minimum phase systems: application to VSTOL aircraft. *Automatica*, 28(4):665–679, 1992.

[13] Masha Ishutkina. Design and implimentation of a supervisory safety controller for a 3DOF helicopter. Master's thesis, Massachusetts Institute of Technology, 2004.

[14] Luis O. Mejias, Srikanth Saripalli, Pascual Cervera, and Gaurav S. Sukhatme. Visual servoing of an autonomous helicopter in urban areas using feature tracking. *Journal for Field Robotics*, 2006.

[15] Richard M. Murray. *A Mathematical Introduction to Robotic Manipulation.* CRC, 1994.

[16] *Pointgrey Website.* www.ptgrey.com, last accessed Feb 2006.

[17] *Quanser Website.* www.quanser.com, last accessed Feb 2006.

[18] Shankar Sastry. *Nonlinear Systems Analysis, Stability and Control.* Springer, 1999.

[19] J.J.E. Slotine and W. Li. *Applied Nonlinear Control.* Pearson Education, 1990.

[20] Salah Sukkarieh. *Low Cost, High Integrity, Aided Inertial Navigation Systems for Autonomous Land Vehicles.* PhD thesis, University of Sydney, Sdyney,Australia, March 2000.