Institutionen för datavetenskap
Department of Computer and Information Science

Master's Thesis

Autonomous UAV Landing Using Monocular Vision in GPS-denied Environments

Jonatan Olofsson

Reg Nr: LiTH-ISY-EX--YY/XXXX--SE
Linköping YYYY

Linköping University
INSTITUTE OF TECHNOLOGY

Department of Computer and Information Science
Linköpings universitet
SE-581 83 Linköping, Sweden

Institutionen för datavetenskap
Department of Computer and Information Science

Master's Thesis

Autonomous UAV Landing Using Monocular Vision in GPS-denied Environments

Jonatan Olofsson

Reg Nr: LiTH-ISY-EX--YY/XXXX--SE
Linköping YYYY

Supervisor: **Rudol, Piotr**
IDA, Linköpings universitet
Schön, Thomas
ISY, Linköpings universitet
Wzorek, Mariusz
IDA, Linköpings universitet

Examiner: **Doherty, Patrick**
IDA, Linköpings universitet

Department of Computer and Information Science
Linköpings universitet
SE-581 83 Linköping, Sweden

Avdelning, Institution Division, Department Department of Computer and Information Science Department of Computer and Information Science Linköpings universitet SE-581 83 Linköping, Sweden	**Datum** Date YYYY-09-19

Språk Language ☐ Svenska/Swedish ☒ Engelska/English ☐ ________	**Rapporttyp** Report category ☐ Licentiatavhandling ☒ Examensarbete ☐ C-uppsats ☐ D-uppsats ☐ Övrig rapport ☐ ________	**ISBN** — **ISRN** LiTH-ISY-EX--YY/XXXX--SE **Serietitel och serienummer** Title of series, numbering **ISSN** —

URL för elektronisk version

http://www.ida.liu.se/divisions/aiics/

http://www.ida.liu.se/divisions/aiics/

Titel
Title Autonomous UAV Landing Using Monocular Vision in GPS-denied Environments

Författare
Author Jonatan Olofsson

Sammanfattning
Abstract

This thesis treats the subject of landing a quadrocopter autonomously.

Traditionally this, and most advanced UAV control, has been approached using off-board cameras and sensors to get a precise pose estimation. This however restricts the utility of the controlled UAV's to a very limited space where sophisticated and expensive gear is available and properly configured. By using only on-board sensors, not only is the utility of the UAV greatly increased, but costs are also cut.

To assist the positioning the quadrotor is equipped with a camera utilized by a library for monocular SLAM.

Nyckelord
Keywords automated landing, slam, ptam, control, uav

Abstract

This thesis treats the subject of landing a quadrocopter autonomously.

Traditionally this, and most advanced UAV control, has been approached using off-board cameras and sensors to get a precise pose estimation. This however restricts the utility of the controlled UAV's to a very limited space where sophisticated and expensive gear is available and properly configured. By using only on-board sensors, not only is the utility of the UAV greatly increased, but costs are also cut.

To assist the positioning the quadrotor is equipped with a camera utilized by a library for monocular SLAM.

Sammanfattning

Svenskt abstract kan man placera här.

Acknowledgments

Till mor och far

Contents

Chapter 1

Introduction

The goal for this thesis is to develop and implement an autonomous landing mode for the quadrotor developed by the AIICS team at Linköping University, the LinkQuad.

The size of the system poses limitations on the payload and processing power available in-flight. This means that the implementations has to be done in an efficient manner on the small computers that are available on the LinkQuad.

The limitations, however, do not stop us from utilizing advanced estimation and control techniques, and it turns out that the LinkQuad design is in fact very suitable for the camera-based pose estimation. This estimation can be efficiently detached from the core control and modularized as an independent sensor, which will be exploited in the thesis.

1.1 Unmanned Aerial Vehicles

As noted in [19], Unmanned Aerial Vehicles (UAV's) have been imagined and constructed for millennia, starting in ancient Greece and China. The modern concept of UAV's was introduced in the first world war, which illuminates the dominant role that the military has played in the field over the last century. A commonly cited alliteration is that UAV's are intended to replace human presence in missions that are ”Dull, Dirty and Dangerous”.

While the military[13] continue to lead the development in the field, recent years have seen a great increase in domestic and civilian applications[22]. These applications range from pesticide dispersing and crop monitoring to traffic control, border watch and rescue scenarios[3].

The type of UAV that is used in the implementation of this thesis falls under the category of Small Unmanned Aerial Vehicles (SUAV's). SUAV's are designed to be man-portable in weight and size and is thus limited in payload and availabe processing power. This limitation, in combination with the unavailability of indoor GPS positioning, has led to extensive use of off-board positioning and control in recent research. Systems developed by for instance Vicon[1] and Qualisys[2] yield positioning with remarkable precision, but they also limit the application to a confined environment with an expensive setup.

This thesis seeks a different approach, with an efficient self-contained on-board implementation. GPS and external cameras are replaced by inertial sensors and an on-board camera which uses visual SLAM to position the LinkQuad relative to its surroundings.

1.2 The Platform

The LinkQuad is a modular quadrotor developed at Linköping University. The core configuration is equipped with standard MEMS sensors (accelerometers, gyroscopes and a magnetometer), but for our purposes we have also mounted a monocular camera which feeds data into a microcomputer specifically devoted to the processing of the camera feed. This devoted microcomputer is what allows the primary on-board microcomputer to focus on state estimation and control, without beeing overloaded by video processing.

The primary microcomputer is running a framework named C++ Robot Automation Platform (CRAP), which was developed by the thesis' author for this purpose. CRAP is a light-weight automation platform with a purpose similar to that of ROS[3]. It is, in contrast to ROS, primarily designed to run on the kind relatively low-end Linux systems that fits the payload and power demands of a SUAV. The framework is further described in Appendix **??**.

Using the framework, the functionality of the implementation is distributed in separate modules.

Observer Sensor fusing state estimation. Chapter 2.

Control Affine Quadratic Control. Chapter 3.

Logic State-machine for scheduling controller parameters and reference trajectory. Chapter 5.

1.3 Previous work

Here, previous known similar work is referenced and discussed. Previous work can be used for both reference implemetations, but also to recognize limitations and restrictions posed on the systems studied.

The problem of positioning an unmanned quadrotor using visual feedback is a problem which was only fairly recently solved [1, 21]. Few have attempted to use on-board sensors only, but have relied on external setups to track the motions of the quadrotors - admittedly with high precision. Fewer still have succeeded using strictly real-time algorithms running on the limited processing power that standard

[1]`http://www.vicon.com/`
[2]`http://www.qualisys.com/`
[3]`http://www.ros.org/`

UAV's of the size of the LinkQuad generally are equipped with. The LinkQuad is, with its distributed processing power, well suited for a full implementation and solution to the problem.

1.3.1 Autonomous landing

The problem of landing a quadrocopter can to a large extent be boiled down to achieving good pose estimates using available information. This problem is studied in e.g. [12, 2], but the most interesting results are obtained in [1, 21], where the ideas from [9] of using monocular SLAM are implemented on a UAV platform and excellent results are obtained.

[2] implements a landing control reference scheme which is summarized in **??**.

1.3.2 vSLAM

A first take on a vSLAM algorithm is presented in [8], resting on the foundation of a Rao-Blackwellized particle filter with Kalman filter banks. The algorithm presented in [8] also extends to the case of multiple cameras, which could be interesting in a longer perspective. An implementation has been made in a ROS project[14], which may be used for reference.

[9] uses an alternative approach and splits the tracking problem of SLAM from the mapping problem. This means that the mapping can run more advanced algorithms at a slower pace than required by the tracking.

The algorithm proposed in [9] uses selected keyframes from which offsets are calculated continously in the tracking thread, while the mapping problem is addressed separately as fast as possible using information only from these keyframes. This opposed to the traditional vSLAM filtering solution where each frame has to be used for continuous filter updates.

By for instance not considering uncertainties in either camera pose or feature location, the complexity of the algorithm is reduced and the number of studied points can be increased to achieve better robustness and performance[18] than when a filtering solution is used. Again, this is the method on which [21] bases its implementation.

1.4 Objectives

The main objective for the thesis is to perform autonomous landing with the LinkQuad. To achieve this,

1.5 Contributions

None

1.6 Thesis Outline

Chapter 2

State Estimation

A central part of automatic control is to know the state of the device you are controlling. The system studied in this thesis - the LinkQuad - is in constant motion, so determining the up-to-date position if of vital importance to the performance of the control. This chapter deals with the estimation of the states relevant for positioning and controlling the LinkQuad. Filter theory and notation is established in Section 2.1.

In this thesis, an Unscented Kalman Filter (UKF) is used, which extends the linear Kalman filter theory to a non-linear model in an appealing black-box way. The theory of the UKF is treated in Section 2.2.

The motion model of the system is derived and discussed in Section 2.3.

The motions of the system is also captured by the on-board sensors. A measurement z is related to the motion model by the sensor model h;

$$z(t) = h(x(t), u(t), t) \tag{2.1}$$

The models for the sensors used on the LinkQuad are discussed in Section 2.4.

2.1 The Filtering Problem

The problem of estimating the state of a system - in this case it's position, orientation, velocity etc. - is in the Kalman filter framework expressed as the problem of finding the state estimate $\hat{x}$ that in a well defined best way (e.g. with Minimum Mean Square Error, MMSE) describes the behaviour of the system.

The evolution of a system plant is traditionally described by a set of differential equations that link the change in the variables to the current state and known inputs, u. The system is also assumed to be subject to an additive white Gaussian noise $v(t)$ with known covariance Q. This introduces an uncertainty associated with the system, which accounts for imperfections in the model compared to the physical real-worl-system.

$$\dot{x}(t) = f_c(x(t), u(t), t) + v_c(t) \tag{2.2}$$

With numeric or analytical solutions, we can obtain the discrete form of (2.2), where only the sampling times are considered. The control signal, $u(t)$, is for instance assumed to be constant in the time interval, and we obtain obtain the next predicted state directly, yielding the prediction of $\hat{x}$ at the time t *given* the information at time $t-1$. [1] This motivates the notation used in this thesis - $\hat{x}_{t|t-1}$.

$$x_{t|t-1} = f(x_{t-1|t-1}, u_t, t) + v(t) \tag{2.3}$$

In the ideal case, a simulation of a prediction $\hat{x}$ would with the prediction model in (2.3) fully describe the evolution of the system. To be able to provide a good estimate in the realistic case, however, we must also feed back measurements given from sensors measuring properties of the system.

These measurements, y_t, are fed back and fused with the prediction using the *innovation*, ν.

$$\nu_t = y_t - \hat{y}_t \tag{2.4}$$

That is, the difference between the measured value and what would be expected in the ideal (simulated) case. To account for disturbances affecting the sensors, the measurements are associated with an additive white Gaussian noise $w(t)$, with known covariance R.

$$\hat{y}_t = h(\hat{x}_t, u_t, t) + w(t) \tag{2.5}$$

The innovation is then fused with the prediction to yield a new estimation [5] of x given the information available as of the time t.

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t \nu_t \tag{2.6}$$

The choice of K_t is a balancing between of trusting the model, or trusting the measurement. In the Kalman filter framework, this balancing is made by tracking and weighing the uncertainties introduced by the prediction and the measurement noise.

Because of the assumptions on the noise and the linear property of the innovation feedback, the Gaussian property of the noise is preserved in the filtering process. The system states can thus ideally be considered drawn from a normal distribution.

$$x \sim \mathcal{N}(\hat{x}, P_{xx}) \tag{2.7}$$

Conditioned on the state and measurements before time k, the covariance of the sample distribution is defined as

$$P_{xx}(t|k) = E\left[\left\{x(t) - \hat{x}_{t|k}\right\}\left\{x(t) - \hat{x}_{t|k}\right\}^T |\mathcal{Z}^j\right]. \tag{2.8}$$

As new measurements are taken, the covariance of the state evolves [7] with the state estimate as

$$P_{xx}(t|t) = P_{xx}(t|t-1) - K_t P_{\nu\nu}(t|t-1) K_t^T \tag{2.9}$$

[1]Note that we have not yet performed any measurements that provide information about the state at time t.

$$P_{\nu\nu}(t|t-1) = P_{yy}(t|t-1) + R(t). \tag{2.10}$$

K is derived in e.g. [5] as

$$K_t = P_{xy}(t|t-1)P_{\nu\nu}^{-1}(t|t-1), \tag{2.11}$$

given the cross-correlation of the predicted state and output, P_{xy}.

There are several approaches to how to propagate the covariance through the prediction-model to acquire $P_{xx}(t|t-1)$ with retained Gaussian properties. As the linear case is simple;

$$P_{xx}(t|t-1) = AP(t|t)A^T + Q_t; \tag{2.12}$$

a novel approach is to linearize the system for A. This linearization, however, fails to capture the finer details of highly non-linear systems and may furthermore be tedious to calculate, analytically or otherwise. A different approach is discussed in Section 2.2.

2.2 The Unscented Kalman Filter

The basic version of the Unscented Kalman Filter was proposed in [7] based on the following intuition [7]

> *With a fixed number of parameters it should be easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function.*

The approach is thus to propagate the uncertainty of the system through the non-linear system and fit the results as a Gaussian distribution. The propagation is made by simulating the system in the prediction model for carefully chosen offsets from the current state called *sigma points*, each associated with a weight of importance. The selection scheme for these points can vary (and yield other types of filters), but a common choice is the *Scaled Unscented Transform* (SUT) [20]. The SUT uses a minimal set of sigma points needed to describe the first two moments of the propagated distribution - two for each dimension (n) of the state vector and one for the mean.

$$\begin{aligned}
\mathcal{X}_\mathbf{0} &= \hat{x} \\
\mathcal{X}_\mathbf{i} &= \hat{x} + \left(\sqrt{(n+\lambda)P_{xx}}\right)_i & i &= 1, \cdots, n \\
\mathcal{X}_\mathbf{i} &= \hat{x} - \left(\sqrt{(n+\lambda)P_{xx}}\right)_i & i &= n+1, \cdots, 2n
\end{aligned} \tag{2.13}$$

$$\begin{aligned}
W_0^m = \tfrac{\lambda}{n+\lambda} \quad W_0^c &= \tfrac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta) \\
W_i^m = W_i^c &= \tfrac{1}{2(n+\lambda)} \qquad i = 1, \cdots, 2n
\end{aligned} \tag{2.14}$$

$$\lambda = \alpha^2(n+\kappa) - n \tag{2.15}$$

Variable	**Value**	**Description**
α	$0 \leq \alpha \leq 1$ (e.g. 0.01)	Scales the size of the sigma point distribution. A small α can be used to avoid large non-local non-linearities.
β	2	As discussed in [6], β affects the weighting of the center point, which will directly influence the magnitude of errors introduced by the fourth and higher order moments. In the strictly Gaussian case, $\beta = 2$ can be shown to be optimal.
κ	0	κ is the number of times that the centerpoint is included in the set of sigma points, which will add weight to the centerpoint and scale the distribution of sigma points.

Table 2.1. Description of the parameters used in the SUT.

The three parameters introduced here, α, β and κ are summarized in table 2.2. The term $\left(\sqrt{(n+\lambda)P_{xx}}\right)_i$ is used to denote the i'th column of the matrix square root $\sqrt{(n+\lambda)P_{xx}}$.

When the sigma points $\mathcal{X}_\mathbf{i}$ has been calculated, they are propagated through the non-linear prediction function and the resulting mean and covariance can be calculated.

$$\mathcal{X}_\mathbf{i}^+ = f(\mathcal{X}_\mathbf{i}, u, t) \qquad i = 0, \cdots, 2n \tag{2.16}$$

$$\hat{x} = \sum_{i=0}^{2n} W_i^m \mathcal{X}_\mathbf{i}^+ \tag{2.17}$$

$$P_{xx} = \sum_{i=0}^{2n} W_i^c \left\{\mathcal{X}_\mathbf{i}^+ - \hat{y}\right\} \left\{\mathcal{X}_\mathbf{i}^+ - \hat{y}\right\}^T \tag{2.18}$$

For the measurement update, similar results are obtained, and the equations (2.21)-(2.22) can be connected to equations (2.10)-(2.11).

$$\mathcal{Y}_\mathbf{i} = h(\mathcal{X}_\mathbf{i}, u, t) \qquad i = 0, \cdots, 2n \tag{2.19}$$

$$\hat{y} = \sum_{i=0}^{2n} W_i^m \mathcal{Y}_\mathbf{i} \tag{2.20}$$

$$P_{yy} = \sum_{i=0}^{2n} W_i^c \left\{\mathcal{Y}_\mathbf{i} - \hat{y}\right\} \left\{\mathcal{Y}_\mathbf{i} - \hat{y}\right\}^T \tag{2.21}$$

$$P_{xy} = \sum_{i=0}^{2n} W_i^c \left\{\mathcal{X}_\mathbf{i} - \hat{x}\right\} \left\{\mathcal{Y}_\mathbf{i} - \hat{y}\right\}^T \tag{2.22}$$

As can be seen from the equations in this section, the UKF handles the propagation of the probability densities through the model without the need for explicit calculation of the Jacobians of Hessians for the system. The filtering is based solely on function evaluations of small offsets from the expected mean state, be it for the measurement functions, discussed in Section 2.4, or the time update prediction function - the motion model.

2.3 Motion Model

As the system studied in the filtering problem progresses through time, the state estimate can be significantly improved by if a prediction is made on what measurements can be expected, and evaluating the plausibility of each measurement after how well they correspond to the prediction. With assumed Gaussian white noise distributions, this evaluation can be done in the probabilistic Kalman framework as presented in Sections 2.1-2.2, where the probablity estimate of the sensors' measurements are based on the motion model's prediction. In this section, a motion model is derived and evaluated.

2.3.1 Reference frames

In the model of the quadrotor, there are several frames of reference.

North-East-Down Earth Fixed (NEDEF) This frame is fixed at a given origin in the earth and is considered an inertial frame. All states are expressed in this frame of reference unless explicitly stated otherwise.

North-East-Down (NED) The NED-frame is fixed at the center of gravity of the quadrotor. The NED system's $\hat{z}$-axis is aligned with the gravitational axis and the $\hat{x}$-axis along the northern axis. The $\hat{y}$-axis is chosen to point east to form a right-hand system.

Body-fixed The body-fixed coordinate system is fixed in the quadrotor with $\hat{x}$-axis in the forward direction and the z-axis in the downward direction - as depicted in Figure **??**.

Propeller fixed Each of the propellers are associated with their own frame of reference, P_i, which tracks the virtual tilting of the thrust vector due to flapping, discussed in Section 2.3.3.

Camera frame This is the frame which describes the location of the camera.

IMU frame This is the body-fixed frame in which the IMU measurements are said to be done. The origin is thus fixed close to the interial sensors.

The conversion between the reference frames are characterized by a transformation including translation and a three-dimensional rotation. Both the origin of the body-centered reference frames - the quadrotor's position - and the rotation of the body-fixed system are stored as system states.

The centers of each of the propeller fixed coordinate systems are parametrized on the height h and distance d from the center of gravity as follows

$$D_0 = (d, 0, h)^{BF} \tag{2.23}$$
$$D_1 = (0, -d, h)^{BF} \tag{2.24}$$
$$D_2 = (-d, 0, h)^{BF} \tag{2.25}$$
$$D_3 = (0, d, h)^{BF} \tag{2.26}$$

Notation: In the following sections, vectors and points in e.g. the NED coordinate systems are denoted x^{NED}. Rotation described by unit quaternions are denoted $R(q)$ for the quaternion q, corresponding to the matrix rotation [10] given by

$$\begin{pmatrix} q_1^2 + q_i^2 - q_j^2 - q_k^2 & 2q_iq_j + 2q_1q_k & 2q_iq_k - 2q_1q_j \\ 2q_iq_j - 2q_1q_k & q_1^2 - q_i^2 + q_j^2 - q_k^2 & 2q_jq_k + 2q_1q_i \\ 2q_iq_k + 2q_1q_j & 2q_jq_k - 2q_1q_i & q_1^2 - q_i^2 - q_j^2 + q_k^2 \end{pmatrix} \tag{2.27}$$

2.3.2 Kinematics

The motions of the quadrotor are described by the following relations [15]:

$$\dot{\xi} = R(q^c)V \tag{2.28a}$$

$$\begin{pmatrix} \dot{q}_1^c \\ \dot{q}_i^c \\ \dot{q}_j^c \\ \dot{q}_k^c \end{pmatrix} = -\frac{1}{2} \begin{pmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & -\omega_z & \omega_y \\ \omega_y & \omega_z & 0 & -\omega_x \\ \omega_z & -\omega_y & \omega_x & 0 \end{pmatrix} \begin{pmatrix} q_1^c \\ q_i^c \\ q_j^c \\ q_k^c \end{pmatrix} \tag{2.28b}$$

In practice, a normalization step also has to be added to account for the unit length constraint on rotation quaternions.

2.3.3 Dynamics

The motions of the quadrotor can be fully mathematically explained by the forces and moments acting on the vehicle. Using the rigid-body assumtion, Eulers' extension of Newton's laws of motion for the quadrotor's center of gravity, $\mathcal{G}$, yields

$$\dot{V} = a_{\mathcal{G}} = \frac{1}{m}\sum F \tag{2.29a}$$

$$\dot{\omega} = I_{\mathcal{G}}^{-1}\sum M_{\mathcal{G}} \tag{2.29b}$$

The main forces acting upon the quadrotor are the effects of three different components

$\sum_{i=0}^{3} F_{ri}$ Rotor thrust,

F_g Gravity,

F_{wind} Wind.

Of these, the gravity is trivially described as

$$F_g = mg \cdot e_3^{NED} \tag{2.30}$$

The following sections will describe the rotor thrust and wind forces respectively.

Rotor thrust

Each of the four propellers on the quadrotor induce a torque and a thrust vector on the system. Due to the differences in relative airspeed around the rotor blade tip as the blades move either along or against the wind-relative velocity, the direction of the thrust will vary with regards to the motions of the quadrotor.

This phenomenon is called *flapping*, and as discussed in e.g. [15], the flapping of the rotors and the centrifugal force acting upon the rotating blades will result in that the tilted blade trajectories will form a cone with the plane to which the rotor axis is normal. These motions of the propellers add important dynamics to the description of the quadrotors motion.

It is desirable, for the purpose of this this thesis and considering computational load, to find a closed-form solution to the flapping equations. This implies several approximations and restrictions which will be discussed in 6.1. The resulting flapping angles and their impact on the thrust vectors can be described as in equations (2.31c)-(2.32) [15, 16, 11].

The momentums induced by the propeller rotation and thrust are described in equations (2.31a)-(2.31b).

$$Q_i = -C_Q \rho A R^3 \omega_{ri} ||\omega_{ri}|| e_3^{NED} \tag{2.31a}$$

$$M_{ri} = F_{ri} \times D_{ri} \tag{2.31b}$$

$$F_{ri}^{BF} = C_T \rho A_r R^2 \omega_{ri}^2 \begin{pmatrix} -\sin(a_{1_s i}) \\ \cos(a_{1_s i}) \sin(b_{1_s i}) \\ -\cos(a_{1_s i}) \cos(b_{1_s i}) \end{pmatrix} \tag{2.31c}$$

The equations for the flapping angles $(a_{1_s i}, b_{1_s i})$ are derived in [15, 16, 11], but are in (2.32) extended to include the velocity relative to the wind. $V_{ri(n)}$ denotes the n'th element of the vector V_{ri}.

$$V_{ri} = V_{rel} + \Omega \times D_{ri} \tag{2.32a}$$

$$\mu_{ri} = \frac{||V_{ri(1,2)}||}{w_i R} \tag{2.32b}$$

$$\psi_{ri} = \arctan\left(\frac{V_{ri(2)}}{V_{ri(1)}}\right) \tag{2.32c}$$

$$\begin{pmatrix} a_{1_s}i \\ b_{1_s}i \end{pmatrix} = \begin{pmatrix} \cos(\psi_{ri}) & -\sin(\psi_{ri}) \\ \sin(\psi_{ri}) & \cos(\psi_{ri}) \end{pmatrix} \begin{pmatrix} \frac{1}{1-\frac{\mu_{ri}^2}{2}}\mu_{ri}\left(4\theta_{twist} - 2\lambda_i\right) \\ \frac{1}{1+\frac{\mu_{ri}^2}{2}}\frac{4}{3}\left(\frac{C_T}{\sigma}\frac{2}{3}\frac{\mu_{ri}\gamma}{a} + \mu_{ri}\right) \end{pmatrix} + \begin{pmatrix} \frac{-\frac{16}{\gamma}\left(\frac{\omega_\theta}{\omega_{ri}}\right)+\left(\frac{\omega_\psi}{\omega_{ri}}\right)}{1-\frac{\mu_{ri}^2}{2}} \\ \frac{-\frac{16}{\gamma}\left(\frac{\omega_\psi}{\omega_{ri}}\right)+\left(\frac{\omega_\theta}{\omega_{ri}}\right)}{1+\frac{\mu_{ri}^2}{2}} \end{pmatrix} \tag{2.32d}$$

Note: In the equations (2.32d)**, taken from [15], I assume that by "**a**", they mean lift curve slope, and by "**a_0**", they mean the linearization point of a and NOT the coning angle of Prouty pp.468, or the mean coning of Prouty pp.153**

[15] [16] pp. 165 **Not quite finished here..**

$$C_T = \frac{\sigma a}{4}\left[\theta_{tip} - \right] \tag{2.32e}$$

Wind

For describing the wind's impact on the quadrotor motion, a simple wind model is applied where the wind is modelled with a static velocity that imposes forces and moments on the quadrotor. The wind velocity vector is estimated by the UKF and may thus still vary in its estimation trough the measurement update. The wind velocities in the filter is given in the NED reference frame.

The wind drag force is calculated using equation (2.33), whereas the moments are given by equations (2.34). In this thesis, the moments acting on the quadrotor body (as opposed to the rotors) are neglected or described by moments imposed by the wind acting on the rotors.

$$F_{wind} = F_{wind,body} + \sum_{i=0}^{3} F_{wind,ri} \tag{2.33a}$$

$$F_{wind,body} = -\frac{1}{2}C_D \rho A V_{rel}||V_{rel}|| \tag{2.33b}$$

$$F^{BF}_{wind,ri} = -\frac{1}{2}\rho C_{D,r}\sigma A_r(V_{ri}\cdot e^{BF}_{P_{ri}3})||V_{ri}\cdot e^{BF}_{P_{ri}3}||e^{BF}_{P_{ri}3} \tag{2.33c}$$

$$M_{wind} = M_{wind,body} + \sum_{i=0}^{3} M_{wind,ri} \tag{2.34a}$$

$$M_{wind,body} \approx 0 \tag{2.34b}$$

$$M_{wind,ri} = D^{BF}_{ri} \times F^{BF}_{wind,ri} \tag{2.34c}$$

2.4 Sensor Models

Symbol	Expression	Description	Unit
a	$\frac{\mathrm{d}C_L}{\mathrm{d}\alpha}$	Slope of the lift curve.	$\frac{1}{rad}$
a	2π	Lift curve slope.	$\frac{1}{rad}$
α	-	Propeller angle of attack.	rad
A_r	-	Rotor disk area.	m^2
c	-	Blade chord - the (mean) length between the trailing and leading edge of the propeller.	m
C_L	-	Coefficient of lift.	1
C_{T0}	-	Linearization point for thrust coefficient.	1
C_Q	$C_Q = C_P \approx \frac{C_T^{3/2}}{\sqrt{2}}$ [2]	Torque coefficient, approximated using hovering conditions.	1
γ	$\frac{\rho a c R^4}{I_b}$	γ is the Lock Number [11], described as the ratio between the aerodynamic forces and the inertal forces of the blade.	1
I_b	-	Rotational inertia of the blade	kgm^2
λ_i	$\sqrt{C_T/2}$	λ_i denotes the inflow to the propeller, which is approximated by the expression to the left.	1
R	-	Rotor radius.	m
ρ	-	Air density.	$\frac{kg}{m^3}$
σ	$\frac{\text{blade area}}{\text{disk area}}$	Disk solidity.	1
θ_{twist}	-	The angle with which the propeller is twisted.	rad

Table 2.2. Table of symbols used in the flapping equations

Symbol	Expression	Description
A	-	3x3 matrix describing the area of the quadrotor, excluding the rotors.
C_D	-	3x3 matrix describing the drag coefficients of the quadrotor.
C_{Dr}	-	Propeller's coefficient of drag.

Table 2.3. Table of symbols used in the wind equations

Chapter 3

Controller

To control the quadrotor's movements, a controller is applied to the physical system, using a model of the system to calculate the best (in a sense well defined in this chapter) signals of control to each of the engines driving the propellers.

The controller approach chosen in this thesis is based on the Linear Quadratic (LQ) controller, the theory of which is presented in Section 3.1. An extension to the technique of *gain-scheduling* is discussed in Section 3.2.

The physical model of the system was derived in Section 2.3. In Section 3.3, this is further developed and adapted for compatibility as a model for the controller.

3.1 The Linear Quadratic Controller

The basic LQ controller, described in e.g. [4], uses a linear state-space system model and weights on the states (Q) and control signals (R) respectively to calculate the control signals that would minimize the integral

$$\mathcal{J} = \int_0^\infty e^T(t)Qe(t) + u^T(t)Ru(t)dt. \tag{3.1}$$

Thus, by varying the elements of the cost matrices Q and R respectively, the solution to the optimization will yield control signals that will steer the system in a fashion that the amplitude of the control signals and the errors are balanced. By e.g. increasing the costs of the control signals, the system LQ controller will issue smaller control signals, protecting the engines but slowing the system down.

In the linear case, (3.1) can be solved analytically, resulting in a linear feedback

$$u_t = -L\hat{x}_t + L_r r_t \tag{3.2}$$

$$L = R^{-1}B^T S, \tag{3.3}$$

where S is the Positively Semi-Definite (PSD) solution to the Continuous Algebraic Riccati Equation (CARE)[4],

$$A^T S + SA + M^T QM - SBR^{-1}B^T S = 0. \tag{3.4}$$

To improve the reference following abilities of the controller, the reference is brought into the control signal by a scaling matrix L_r, which is chosen so that the static gain of the system is equal to identity [4]. In the case of equal number of control signals as controlled states, the following result is obtained;

$$L_r = \left[M(BL - A)^{-1}B\right]^{-1}. \tag{3.5}$$

3.2 LQ Gain-Scheduling

Even though any system could be described at any point by its linearization, the linear nature of the LQ control poses a limitation in that a general system such as the one studied in this thesis - the LinkQuad - will sooner or later leave the vicinity of the linearization point and no longer adhere to the physical circumstances of the linearization point.

This will lead to sub-optimal control and possibly even to system failure. A common approach is to switch between pre-calculated control gains which has been calculated for selected linearization points.

The approach used in this thesis is closely related to gain scheduling, but instead of using pre-calculated gains, the linearization is done in-flight at each time instant from the analytical expression of the system's Jacobian.

$$\dot{x} = f(x,u) \approx f(x_0,u_0) + \underbrace{\left.\frac{\partial f}{\partial x}\right|_{\substack{x = x_0 \\ u = u_0}}}_{A} \underbrace{(x - x_0)}_{\Delta x} + \underbrace{\left.\frac{\partial f}{\partial u}\right|_{\substack{x = x_0 \\ u = u_0}}}_{B} \underbrace{(u - u_0)}_{\Delta u} \tag{3.6}$$

In the standard formulation of LQ, the linearization is made around a stationary point (x_0, u_0), where $f(x_0,u_0) = 0$. In a more general formulation, it is possible to lift this constraint using homogenous equations [17];

$$\dot{X} = \begin{bmatrix} \dot{x} \\ * \end{bmatrix} = \underbrace{\begin{bmatrix} A & f(x_0,u_0) \\ 0 & 0 \end{bmatrix}}_{\bar{A}} \underbrace{\begin{bmatrix} \Delta x \\ 1 \end{bmatrix}}_{\bar{X}} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{\bar{B}} \Delta u. \tag{3.7}$$

Equation 3.7 is a linear system for which the ordinary LQ problem can be solved, using eq. (3.2)-(3.5).

Generally the linearized output signal, Δu, would be added to u_0 to form the controller outout, as per

$$u = u_0 + \Delta u = u_0 - L\Delta x + L_r \Delta r.$$

However, the approach to re-linearize the system continously leads to the property that, since the controller will always work at its linearization point, and we can note that Δx will thus always be zero, giving the following expression for the calculation of the control output;

$$u_{t+1} = u_t + \Delta u = u_0 + L_r(r - x). \tag{3.8}$$

3.3 Control Model

In Chapter 2, a physical model of the system was derived. To incorporate the information from the physical model into the governing control law, the model needs to be fitted into (3.7) by providing analytical[1] expressions for the Jacobi matrices with regards to x and y, and removing states which will not be used in the controller.

The model in Chapter 2 should be validated before proceeding with this section.

[1]Analytical results are not strictly needed but are, in this case, easily obtained.

Chapter 4

Video

Chapter 5

Logic

Chapter 6

Discussion

In this chapter the results of the thesis are discussed. Specific attention is given to restrictions and their impact on system performance, as well as providing a basis for later discussions on further work.

6.1 Flapping

The flapping equations pretty much suck.

Chapter 7

Conclusions

7.1 Further work

Wind model, drag model, wind momentum on body

Bibliography

[1] Michael Blösch, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Vision based MAV navigation in unknown and unstructured environments. In *ICRA*, pages 21–28, 2010.

[2] Roland Brockers, Patrick Bouffard, Jeremy Ma, Larry Matthies, and Claire Tomlin. Autonomous landing and ingress of micro-air-vehicles in urban environments based on monocular vision. In Thomas George, M. Saif Islam, and Achyut K. Dutta, editors, *Micro- and Nanotechnology Sensors, Systems, and Applications III*, volume 8031, page 803111. SPIE, 2011.

[3] Patrick Doherty and Piotr Rudol. A UAV Search and Rescue Scenario with Human Body Detection and Geolocalization, 2007.

[4] T. Glad and L. Ljung. *Reglerteori: flervariabla och olinjära metoder*. Studentlitteratur, 2003.

[5] F. Gustafsson. *Statistical Sensor Fusion*. Utbildningshuset/Studentlitteratur, 2010.

[6] Simon J. Julier and Idak Industries. The scaled unscented transformation. In *in Proc. IEEE Amer. Control Conf*, pages 4555–4559, 2002.

[7] Simon J | Uhlmann Jeffrey K | Durrant-Whyte Hugh F Julier. A new approach for filtering nonlinear systems. In *1995 American Control Conference, 14th, Seattle, WA; UNITED STATES; 21-23 June 1995*, pages 1628–1632, 1995.

[8] Niklas Karlsson, Enrico Di Bernardo, Jim Ostrowski, Luis Goncalves, Paolo Pirjanian, and Mario E. Munich. The vSLAM algorithm for robust localization and mapping. In *In Proc. of Int. Conf. on Robotics and Automation (ICRA*, pages 24–29, 2005.

[9] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[10] J.B. Kuipers. *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality*. Princeton paperbacks. Princeton University Press, 2002.

[11] J.G. Leishman. *Principles of helicopter aerodynamics*. Cambridge aerospace series. Cambridge University Press, 2002.

[12] D. Mellinger, M. Shomin, and V. Kumar. Control of Quadrotors for Robust Perching and Landing. In *Proceedings of the International Powered Lift Conference*, Oct 2010.

[13] United States. Dept. of Defense. Office of the Secretary of Defense. *U.S. Army Roadmap for unmanned aircraft systems, 2010-2035*. U. S. Army UAS Center of Excellence, 2010.

[14] Helen Oleynikova. ROS vSLAM, September 2011. [online] `http://www.ros.org/wiki/vslam`.

[15] Paul Pounds, Robert Mahony, and Peter Corke. Modelling and Control of a Quad-Rotor Robot.

[16] R.W. Prouty. *Helicopter performance, stability, and control*. Krieger Pub., 1995.

[17] A. Rantzer and M. Johansson. Piecewise Linear Quadratic Optimal Control, 1999.

[18] Hauke Strasdat, J. M. M. Montiel, and Andrew J. Davison. Real-time monocular SLAM: Why filter? In *ICRA*, pages 2657–2664, 2010.

[19] K. Valavanis. *Advances in unmanned aerial vehicles: state of the art and the road to autonomy*. International series on intelligent systems, control, and automation. Springer, 2007.

[20] Rudolph van der Merwe, Arnaud Doucet, Nando de Freitas, and Eric A. Wan. The Unscented Particle Filter. In *NIPS'00*, pages 584–590, 2000.

[21] S Weiss, D Scaramuzza, and R Siegwart. Monocular-SLAM based navigation for autonomous micro helicopters in GPS-denied environments. *Journal of Field Robotics*, 28(6):854–874, 2011.

[22] K C Wong and C Bil. UAVs OVER AUSTRALIA - Market And Capabilities. *Flight International*, pages 1–16, 2006.