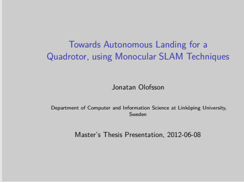
Towards Autonomous Landing for a Quadrotor, using Monocular SLAM Techniques

Jonatan Olofsson

Department of Computer and Information Science at Linköping University, Sweden

Master's Thesis Presentation, 2012-06-08


Towards Autonomous Landing for a Quadrotor, using Monocular SLAM Techniques

Jonatan Olofsson

Department of Computer and Information Science at Linköping University, Sweden

Master's Thesis Presentation, 2012-06-08

Hello and welcome to my thesis presentation. My name is Jonatan Olofsson and have been working on this thesis for the past semester.

I will begin this hour with a presentation of my work; taking about 20 minutes. Next, Karl-Johan Barsk will present the opposition and after that we will open up for questions.

Presentation Outline

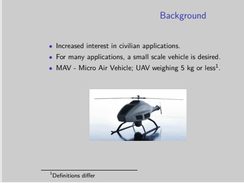
I will begin this presentation with an overview, including a description of the problem I have studied, as well as a short introduction to the techiniques I have studied. I will try to explain what problems I faced and the solutions I applied to make the technologies fit together.

After having described the different parts of the thesis, I will present a few of the main technologies from the thesis in some more detail, namely the monocular SLAM, state estimation and nonlinear control. I will also try to clarify why the application of these techiniques were deemed nescessary to control the descent of the would-be-achieved landing procedure.

Background

- Increased interest in civilian applications.
- For many applications, a small scale vehicle is desired.
- MAV - Micro Air Vehicle; UAV weighing 5 kg or less[1].

[1]Definitions differ



Background

- Increased interest in civilian applications.
- For many applications, a small scale vehicle is desired.
- MAV - Micro Air Vehicle; UAV weighing 5 kg or less[1].

[1]Definitions differ

First a bit of background;

During the past few years, there has been a significant increase in the interest in the field of unmanned aerial vehicles. While the military industry is the major developer of UAVs, the maturity and availability of technology has enabled applications in search-and-rescue operations and even hobby-level implementations.

Many UAVs are of considerable size to carry load and to increase flight-time. As new applications develop however, the interest has increased in small scale UAVs such as the LinkQuad, towards which this thesis have been targeted. This type of UAV is known as MAV, Micro Air Vehicle.

The LinkQuad Platform

Quadrotor research platform developed by AIICS at the Department of Computer and Information Science of Linköping University.

Used sensors: Accelerometers, gyroscopes, pressure sensor, camera

The LinkQuad Platform

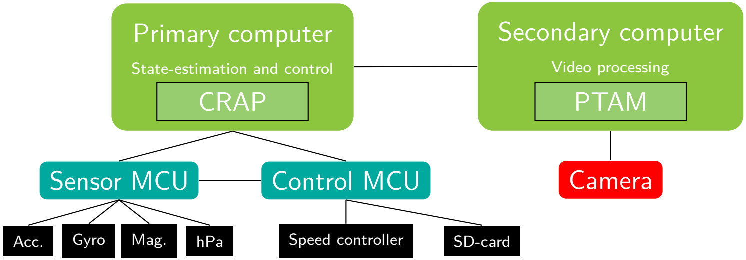
Quadrotor research platform developed by AIICS at the Department of Computer and Information Science of Linköping University.

Used sensors: Accelerometers, gyroscopes, pressure sensor, camera

The LinkQuad platform, which you can see here, is a quadrotor developed here at IDA. It is equipped with a standard set of inertial sensors - accelerometers and gyroscopes - as well as pressure sensors, magnetometers and GPS. It can also be optionally equipped with a camera, which was used in this thesis.

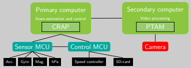
The LinkQuad Platform

- Dual gumstix micro-computers (Linux).
- Sensor-board with dual microcontrollers for sensor sampling, data logging and low-level control.



The LinkQuad Platform

- Dual gumstix micro-computers (Linux).
- Sensor-board with dual microcontrollers for sensor sampling, data logging and low-level control.



One particularly notable feature on the LinkQuad is the dual gumstix Linux microcomputers. This allows us to dedicate one each of these to video processing and state estimation respectively.

These gumstix computers measure only centimeters in size, yet are quite powerful in terms of computational power. They do, however, lack floating point processor units, which in the end was a major limiting factor as to why only simulational flight was achieved.

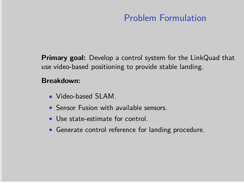
Aside from the gumstix microcomputers, the LinkQuad is also equipped with microcontrollers for sampling the sensors and actuating control signals.

Problem Formulation

Primary goal: Develop a control system for the LinkQuad that use video-based positioning to provide stable landing.

Breakdown:

- Video-based SLAM.
- Sensor Fusion with available sensors.
- Use state-estimate for control.
- Generate control reference for landing procedure.


Problem Formulation

Primary goal: Develop a control system for the LinkQuad that use video-based positioning to provide stable landing.

Breakdown:

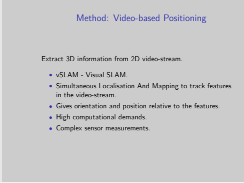
- Video-based SLAM.
- Sensor Fusion with available sensors.
- Use state-estimate for control.
- Generate control reference for landing procedure.

One of the most important features of an independent UAV is the landing procedure. Therefore, the goal of this thesis was to develop a control system that, with the help of camera positioning, could be used to land the LinkQuad.

Method: Video-based Positioning

Extract 3D information from 2D video-stream.

- vSLAM - Visual SLAM.
- Simultaneous Localisation And Mapping to track features in the video-stream.
- Gives orientation and position relative to the features.
- High computational demands.
- Complex sensor measurements.


Method: Video-based Positioning

Extract 3D information from 2D video-stream.

- vSLAM - Visual SLAM.
- Simultaneous Localisation And Mapping to track features in the video-stream.
- Gives orientation and position relative to the features.
- High computational demands.
- Complex sensor measurements.

One of the main technologies that were studied in this thesis is video based SLAM, Simultaneous Localization And Mapping using video-stream data. A vSLAM algorithm - and I say “a”, ’cause there are several, which work in simiilar ways - extracts trackable features from the video images, and tracks their changes in position between the video frames. This is used to extract the camera’s position relative to the features. Video processing generally puts high computational demands, and the resulting measurements are quite complex to interpret. A high-level filtering framework needed to be implemented to integrate the video positioning with the state estimation of the quadrotor.

Method: Filtering

Current implementation is based on complementary filtering.

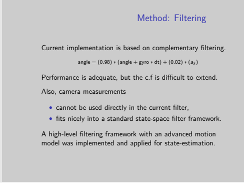
$$\text{angle} = (0.98) * (\text{angle} + \text{gyro} * \text{dt}) + (0.02) * (a_{\hat{x}})$$

Performance is adequate, but the c.f is difficult to extend.

Also, camera measurements

- cannot be used directly in the current filter,
- fits nicely into a standard state-space filter framework.

A high-level filtering framework with an advanced motion model was implemented and applied for state-estimation.



Method: Filtering

Current implementation is based on complementary filtering.

$$\text{angle} = (0.98) * (\text{angle} + \text{gyro} * \text{dt}) + (0.02) * (a_{\hat{x}})$$

Performance is adequate, but the c.f is difficult to extend.

Also, camera measurements

- cannot be used directly in the current filter,
- fits nicely into a standard state-space filter framework.

A high-level filtering framework with an advanced motion model was implemented and applied for state-estimation.

The current observer - that is, implementation of state estimation - is based on the complementary filter, and while the performance of the filter is quite adequate for its current purpose, adding more sensors - especially such as the camera positioning - is quite difficult.

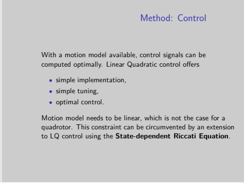
The de facto standard techique for nonlinear state estimation, the Extended Kalman Filter, was ultimately applied for the state estimation. Extensive work was in fact put into creating a physical motion model, and while a simpler model probably would have sufficed for the state estimation, the same model could in fact thereby be re-used later, for instance in the nonlinear control that was implemented.

Method: Control

With a motion model available, control signals can be computed optimally. Linear Quadratic control offers

- simple implementation,
- simple tuning,
- optimal control.

Motion model needs to be linear, which is not the case for a quadrotor. This constraint can be circumvented by an extension to LQ control using the **State-dependent Riccati Equation**.



Method: Control

With a motion model available, control signals can be computed optimally. Linear Quadratic control offers

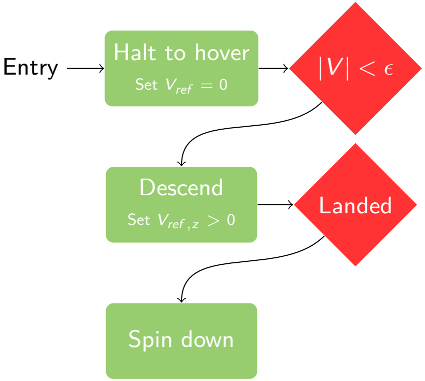
- simple implementation,
- simple tuning,
- optimal control.

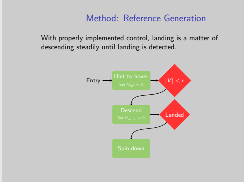
Motion model needs to be linear, which is not the case for a quadrotor. This constraint can be circumvented by an extension to LQ control using the **State-dependent Riccati Equation**.

Having a motion model ment that the LQ control methodology was near at hand to be applied. The problem is, the LQ requires the motion model to be linear. In the thesis, I study a technique to bypass this constraint called State-dependent Riccati Equations, which basically solves the linear problem repeatedly online.

Method: Reference Generation

With properly implemented control, landing is a matter of descending steadily until landing is detected.





Method: Reference Generation

With properly implemented control, landing is a matter of descending steadily until landing is detected.

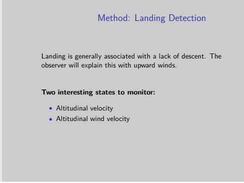
Finally, I also implemented a state-machine engine to generate reference signals for the controlled descent and landing of a quadrotor. The basic idea is to use the positioning for a stabilized descent and to detect the event of landing when the vehicle has hit the ground.

Method: Landing Detection

Landing is generally associated with a lack of descent. The observer will explain this with upward winds.

Two interesting states to monitor:

- Altitudinal velocity
- Altitudinal wind velocity


Method: Landing Detection

Landing is generally associated with a lack of descent. The observer will explain this with upward winds.

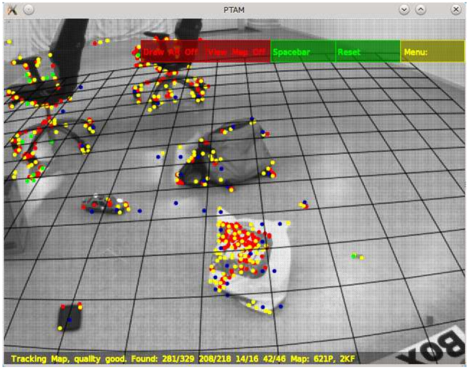
Two interesting states to monitor:

- Altitudinal velocity
- Altitudinal wind velocity

As the quadrotor lands, the observer's current motion model will fail to accurately describe this sudden loss of altitudinal velocity and will instead, after a while's settling time, explain the behaviour of the vehicle by an increased upward force from the modeled wind. Thus, the altitudinal velocity and estimated altitudinal wind velocity could both be monitored for increased detection reliability.

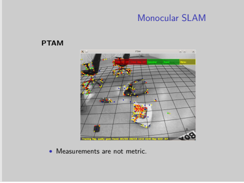
Monocular SLAM

PTAM



Tracking Map, quality good. Found: 281/329 208/218 14/16 42/46 Map: 621P, 2KF

- Measurements are not metric.


Monocular SLAM

PTAM

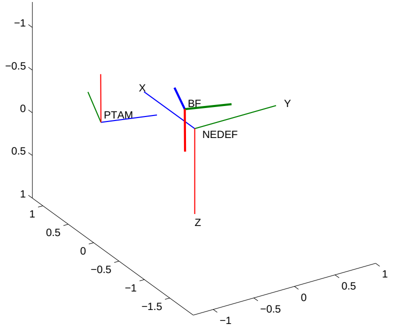
- Measurements are not metric.

For the video based SLAM part of this thesis, a library called PTAM was used. Again, the idea is to track features, here shown as dots, through the video frames to estimate the camera's position. Often, stereo vision is applied to extract depth information from the recorded scene. With monocular SLAM however, the movement of the features need to be related to extract the depth of each feature in the image.

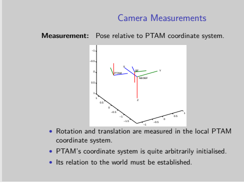
One major problem with video based tracking is that, unless you know something about the recorded scene, there is no way of knowing wether its a large scene that moved a lot, or a small scene that just moved a little. As for instance when you film a dollhouse, there is simply no way to tell it from a real house, and yet the scale of movement is completely different.

Camera Measurements

Measurement: Pose relative to PTAM coordinate system.



- Rotation and translation are measured in the local PTAM coordinate system.
- PTAM's coordinate system is quite arbitrarily initialised.
- Its relation to the world must be established.



Camera Measurements

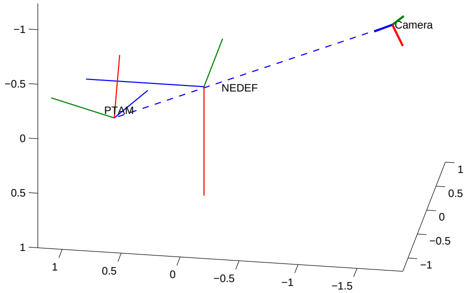
Measurement: Pose relative to PTAM coordinate system.

- Rotation and translation are measured in the local PTAM coordinate system.
- PTAM's coordinate system is quite arbitrarily initialised.
- Its relation to the world must be established.

The scaling needs to be extracted from other sources, and autonomously so since the PTAM coordinate system is initialized quite arbitrarily. This coordinate system is of central importance, since all measurements are given relative to it, and yet both orientation, position and scale is initially unknown. To be able to use the measurements of position and orientation, this transformation needs to be determined using external information.

Initialization

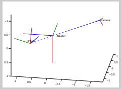
PTAM tries to initialize its coordinate system on ground plane.



$$\begin{cases} \varnothing_{\text{PTAM}} &= \xi + R(q^{wb}) r_{\text{camera}/\mathcal{G}} + \lambda R(q^{wP}) \frac{X^{PTAM}}{|X^{PTAM}|} \\ \varnothing_{\text{PTAM}} \cdot \hat{z} &= 0 \end{cases}$$

Initialization

PTAM tries to initialize its coordinate system on ground plane.



$$\begin{cases} \varnothing_{\text{PTAM}} &= \xi + R(q^{wb}) r_{\text{camera}/\mathcal{G}} + \lambda R(q^{wP}) \frac{X^{PTAM}}{|X^{PTAM}|} \\ \varnothing_{\text{PTAM}} \cdot \hat{z} &= 0 \end{cases}$$

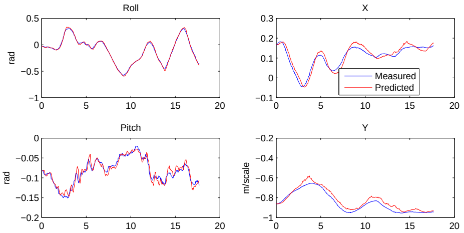
What we do know of the PTAM initialization process is that it tries to place the coordinate system on the ground plane. Knowing our position of initialization, we can calculate backwards from the first measurement and make a guess of the position and orientation of the coordinate system.

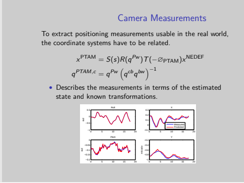
Camera Measurements

To extract positioning measurements usable in the real world, the coordinate systems have to be related.

$$x^{\text{PTAM}} = S(s)R(q^{Pw})T(-\varnothing_{\text{PTAM}})x^{\text{NEDEF}}$$
$$q^{PTAM,c} = q^{Pw}\left(q^{cb}q^{bw}\right)^{-1}$$

- Describes the measurements in terms of the estimated state and known transformations.



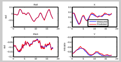


Camera Measurements

To extract positioning measurements usable in the real world, the coordinate systems have to be related.

$$x^{\mathrm{PTAM}} = S(s)R(q^{Pw})T(-\varnothing_{\mathrm{PTAM}})x^{\mathrm{NEDEF}}$$
$$q^{PTAM,c} = q^{Pw}\left(q^{cb}q^{bw}\right)^{-1}$$

- Describes the measurements in terms of the estimated state and known transformations.



The transformation can then be used to connect the incoming measurements to the state estimation framework. As you can see in the equatinons, in the state estimation framework, the measurements are expressed in terms of the available states, not the other way around.

Sensor Fusion

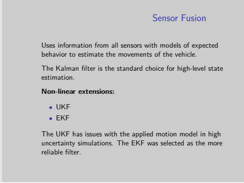
Uses information from all sensors with models of expected behavior to estimate the movements of the vehicle.

The Kalman filter is the standard choice for high-level state estimation.

Non-linear extensions:

- UKF
- EKF

The UKF has issues with the applied motion model in high uncertainty simulations. The EKF was selected as the more reliable filter.



Sensor Fusion

Uses information from all sensors with models of expected behavior to estimate the movements of the vehicle.

The Kalman filter is the standard choice for high-level state estimation.

Non-linear extensions:

- UKF
- EKF

The UKF has issues with the applied motion model in high uncertainty simulations. The EKF was selected as the more reliable filter.

The measurements from the camera need to be fused with the rest of the available sensor data, as well as the motion model. In the linear case, the Kalman filter is provably the optimal bayesian filter given some assumptions, but the theory can also with good results be extended to the nonlinear case. The novel approach is the Extended Kalman filter which simply linearizes the model as nescessary and applies the linear theory. The Unscented Kalman Filter approaches the field of particle filtering by testing variations of the estimated state and look for the best gaussian fit.

In the implementation, this is performed in a generally written sensor fusion module. The current implementation features an Extended Kalman filter, but the Unscented Kalman Filter was also studied in the thesis, but was ultimately replaced due instabilities.

Filtering Framework

The standard formulation of a high-level (Bayesian) state estimation framework relies on two separate steps.

Measurement update: Sample the sensors and weigh their likelihood against the current estimate. **Time update:** Use the motion model to predict the vehicle's motions.

Through discrete instances of time, the steps are independent.

The standard formulation of a high-level (Bayesian) state estimation framework relies on two separate steps.

Measurement update: Sample the sensors and weigh their likelihood against the current estimate. **Time update:** Use the motion model to predict the vehicle's motions.

Through discrete instances of time, the steps are independent.

A filtering framework is generally equipped with two independent steps;

The measurement update step then take the sensorvalues and incorporate them into the state estimate. Since both the state estimate and sensor measurements are associated with noise and uncertainties, these to have to be weighed against each other according the certainty of each.

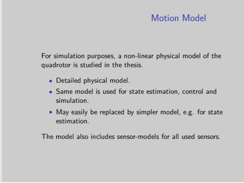
The time update, where the physical model is used to anticipate the movements of the vehicle, based on previous knowledge and reasonable assumptions. One can here, for instace, assume that the velocity is constant, and the vehicle's position must thus be updated according to the known velocity. In this thesis, a physical motion model was developed and presented.

Motion Model

For simulation purposes, a non-linear physical model of the quadrotor is studied in the thesis.

- Detailed physical model.
- Same model is used for state estimation, control and simulation.
- May easily be replaced by simpler model, e.g. for state estimation.

The model also includes sensor-models for all used sensors.


Motion Model

For simulation purposes, a non-linear physical model of the quadrotor is studied in the thesis.

- Detailed physical model.
- Same model is used for state estimation, control and simulation.
- May easily be replaced by simpler model, e.g. for state estimation.

The model also includes sensor-models for all used sensors.

The physical quadrotor model presented in this thesis is in fact developed in some detail, and while some model identification remain, it describes in detail the forces acting on the quadrotor in flight. The physical model is arguably unnescessary complex for many purposes, but the fact that this model actually fits both simulation, state estimation and nonlinear control is a remarkable feature. This does not only add maintainability, but also precision to the state estimation and control. In fact, the structure of the model allows for its re-use in all model-dependent parts of the implementation, removing the need for multiple models to be developed.

Linear Quadratic control

Linear Quadratic Control utilizes a linear motion model to optimally control the system.

Definition: Find $u = -Lx$ s.t.

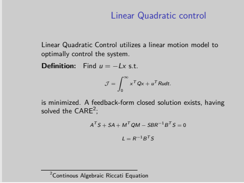
$$\mathcal{J} = \int_0^\infty x^T Q x + u^T R u dt.$$

is minimized. A feedback-form closed solution exists, having solved the CARE[2];

$$A^T S + SA + M^T Q M - SBR^{-1} B^T S = 0$$

$$L = R^{-1} B^T S$$

[2] Continous Algebraic Riccati Equation



Linear Quadratic control

Linear Quadratic Control utilizes a linear motion model to optimally control the system.

Definition: Find $u = -Lx$ s.t.

$$\mathcal{J} = \int_0^\infty x^T Qx + u^T Rudt.$$

is minimized. A feedback-form closed solution exists, having solved the CARE[2];

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

$$L = R^{-1} B^T S$$

[2]Continous Algebraic Riccati Equation

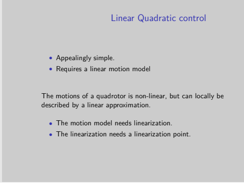
Having aquired a pose estimate, this must be fed to a controller that is able to control the flight and descent of the quadrotor. Linear Quadratic Control is one of the simplest to use optimal control techniques, and makes use of the motion model I just briefly presented. The basic principle of Linear Quadratic Control is to weigh the system response, as predicted by the model, against the cost of using the control signals. That is; using this mathematical criterion, the LQ decides which is better - to rapidly reach the desired velocity, or to be gentle on the motors. An added advantage for the Linear Quadratic Control is that with predetermined control weights - that is, settings for the controller - a feedback solution exists, having solved the Riccati equation exemplified here.

Linear Quadratic control

- Appealingly simple.
- Requires a linear motion model

The motions of a quadrotor is non-linear, but can locally be described by a linear approximation.

- The motion model needs linearization.
- The linearization needs a linearization point.



Linear Quadratic control

- Appealingly simple.
- Requires a linear motion model

The motions of a quadrotor is non-linear, but can locally be described by a linear approximation.

- The motion model needs linearization.
- The linearization needs a linearization point.

However, the Linear Quadratic controller, as the name suggests requires a linear motion model. Problem is that by considering the orientation of the vehicle in the model makes for a nonlinear one, which does not quite fit the original LQ formulation.

One can consider linearising the motion model in various stationary points, although such points are generally difficult to find unless we apply a clever trick.

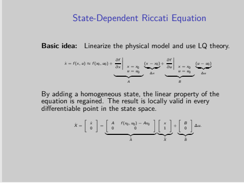
State-Dependent Riccati Equation

Basic idea: Linearize the physical model and use LQ theory.

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

By adding a homogeneous state, the linear property of the equation is regained. The result is locally valid in every differentiable point in the state space.

$$\dot{X} = \begin{bmatrix} \dot{x} \\ 0 \end{bmatrix} = \underbrace{\begin{bmatrix} A & f(x_0,u_0) - Ax_0 \\ 0 & 0 \end{bmatrix}}_{\bar{A}} \underbrace{\begin{bmatrix} x \\ 1 \end{bmatrix}}_{\bar{X}} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{\bar{B}} \Delta u.$$



State-Dependent Riccati Equation

Basic idea: Linearize the physical model and use LQ theory.

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

By adding a homogeneous state, the linear property of the equation is regained. The result is locally valid in every differentiable point in the state space.

$$\dot{X} = \begin{bmatrix} \dot{x} \\ 0 \end{bmatrix} = \underbrace{\begin{bmatrix} A & f(x_0,u_0) - Ax_0 \\ 0 & 0 \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} x \\ 1 \end{bmatrix}}_{\tilde{X}} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{\tilde{B}} \Delta u.$$

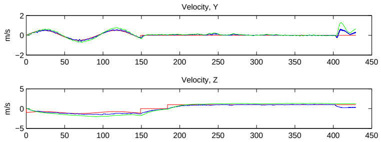
The trick suggested in this thesis, with good results, is to extend the Taylor expansion linearisation with a new virtual state. This virtual state, shown in the second equation, allows us to regain the linear property of an affine system approximation that is the result from a general Taylor expansions, as in the first equation.

By using this general formulation, it is possible to control a general nonlinear system with the same simplicity as in the linear case.

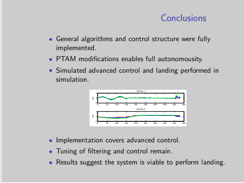
This general case can be solved repeatedly to generate the control signals, a techinique called State-Dependet Riccati Equation which is studied and evaluated in the thesis.

Conclusions

- General algorithms and control structure were fully implemented.
- PTAM modifications enables full autonomousity.
- Simulated advanced control and landing performed in simulation.

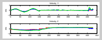


- Implementation covers advanced control.
- Tuning of filtering and control remain.
- Results suggest the system is viable to perform landing.



Conclusions

- General algorithms and control structure were fully implemented.
- PTAM modifications enables full autonomousity.
- Simulated advanced control and landing performed in simulation.



- Implementation covers advanced control.
- Tuning of filtering and control remain.
- Results suggest the system is viable to perform landing.

In the work I present in this thesis covers the implementation and simulational evaluation of the algorithms presented here, as well as a lot of groundwork to connect these in a high-level control system for a quadrotor. I have extended the PTAM monoslam localisation library and incorporated these measurements into an extendable filtering framework to provide the state estimate needed to achieve stable control needed to perform precision landing with a quadrotor.

Results are still simulational, but with continued parameter tuning and enough computational power, they firmly suggest that full control and landing could be performed using the implementation. The work has been focused on implementing state-of-the-art techniques to provide landing and general control for the LinkQuad and, by extension general quadrotors.