Monocular Vision Assisted Autonomous Landing of a Helicopter on a Moving Deck

by

Andre Dewald Swart

Thesis presented in partial fulfilment of the requirements for the degree of
Master of Science in Engineering
at Stellenbosch University

Supervisor:

Dr I.K. Peddle

Department Electrical and Electronic Engineering

March 2013

Stellenbosch University http://scholar.sun.ac.za

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

March 2013

Copyright © 2013 Stellenbosch University
All rights reserved

Stellenbosch University http://scholar.sun.ac.za

Abstract

The landing phase of any helicopter is the most critical part of the whole flight envelope, particularly on a moving flight deck. The flight deck is usually located at the stern of the ship, translating to large heave motions. This thesis focuses on the three fundamental components required for a successful landing: accurate, relative state-estimation between the helicopter and the flight deck; a prediction horizon to forecast suitable landing opportunities; and excellent control to safely unite the helicopter with the flight deck.

A monocular-vision sensor node was developed to provide accurate, relative position and attitude information of the flight deck. The flight deck is identified by a distinct, geometric pattern. The relative states are combined with the onboard, kinematic state-estimates of the helicopter to provide an inertial estimate of the flight deck states. Onboard motion prediction is executed to forecast a possible safe landing time which is conveyed to the landing controller.

Camera pose-estimation tests and hardware-in-the-loop simulations proved the system developed in this thesis viable for flight tests. The practical flight tests confirmed the success of the monocular-vision sensor node.

Stellenbosch University http://scholar.sun.ac.za

Uitreksel

Die mees kritiese deel van die hele vlug-duurte van ’n helikopter is die landings-fase, veral op ’n bewegende vlugdek. Die vlugdek is gewoonlik geleë aan die agterstewe-kant van die skip wat groot afgee bewegings mee bring. Hierdie tesis ondersoek die drie fundamentele komponente van ’n suksesvolle landing: akkurate, relatiewe toestand-beraming tussen die helikopter en die vlugdek; ’n vooruitskatting horison om geskikte landings geleenthede te voorspel; en uitstekended beheer om die helikopter en vlugdek veilig te verenig.

’n Monokulêre-visie sensor-nodus was ontwikkel om akkurate, relatiewe-posisie en oriëntasie informasie van die vlugdek te verwerf. Die vlugdek is geidentifiseer deur ’n kenmerkende, geometriese patroon. Die relatiewe toestande word met die aan-boord kinematiese toestand-afskatter van die helikopter gekombineer, om ’n beraming van die inertiale vlugdek-toestande te verskaf. Aan-boord beweging-vooruitskatting is uitgevoer om moontlike, veilige landingstyd te voorspel en word teruggevoer na die landingsbeheerder.

Kamera-orientasie afskat-toetse en hardeware-in-die-lus simulasies het die ontwikkelde sisteem van hierdie tesis lewensvatbaar vir vlug-toetse bewys. Praktiese vlug-toetse het die sukses van die monokulêre-visie sensor-nodus bevestig.

Stellenbosch University http://scholar.sun.ac.za

Contents

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

List of Figures

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

List of Tables

Stellenbosch University http://scholar.sun.ac.za

Nomenclature

Convention

DCM_{ji}	j^{th} row, i^{th} column element of the DCM matrix
h_{ji}	j^{th} row, i^{th} column element of the Homography matrix

Letters

A	System/Regression Matrix
B	Input Matrix
e	Matching Window Error
J	Cost Function
L	Kalman Filter Gain
P	Propagated State Covariance
H	Output matrix
Q	Quadratic Weighting Matrix, Process Noise Covariance Matrix
R	Autocorrolation Matrix, Measurement Noise Covariance Matrix
y	Measurement, System Output
$\hat{y}$	Linear Predictor Projection, y-Coordinate of Reprojected Image Marker
f	Focal Ratio
c	Principal Offset [Pixels]
k	Radial Distortion Parameter, Discrete Sample Counter
p	Tangential Distortion Parameter

Stellenbosch University http://scholar.sun.ac.za

u	Deterministic Input
w	Linear Predictor Model Coefficient Column
M	Column Order of Linear Predictor Model
N	Row Order of Linear Predictor Model
X	Linear Predictor Model Coefficient Column
x	State vector
$\hat{x}$	x-Coordinate of Reprojected Image Marker

Greek Symbols

ϕ	Roll Angle
θ	Pitch Angle
ψ	Yaw Angle
Λ	Null Space Column
λ	Eigen Value
μ	Learning Rate
σ	Standard Deviation
Φ	Discrete System Matrix
Γ	Null Space Column Normalizing Factor, Discrete Input Matrix

Subscripts

x	x Coordinate Related
y	y Coordinate Related
i	Vector Element corresponding to the i^{th} entry
j,i	Matrix Element corresponding to the j^{th} row and i^{th} row
R	Flight Deck Estimates Coordinated in Camera Lens Axis
C	Camera Lens Coordinated in Helicopter Body Axis

Stellenbosch University http://scholar.sun.ac.za

H	Helicopter Coordinated in Inertial Axis
D	Flight Deck Coordinated in Inertial Axis
north	North Position
east	East Position
down	Down Position
roll	Roll Attitude
pitch	Pitch Attitude
yaw	Yaw Attitude

Abbreviations and Acronyms

2D	Two Dimentional
3D	Three Dimentional
API	Application Programming Interface
AR	Autoregression
ARE	Algebraic Ricatti Equation
ATOL	Automatic Take-off and Landing
AVATAR	Autonomous Vehicle Aerial Tracking And Reconnaissance
BIC	Bayes Information Criterion
bps	Bytes per Second
BRIEF	Binary Robust Independent Elementary Features
BSD	Berkeley Software Distribution
CAN	Controller Area Network
CEP	Circular Error Probable
CG	Center of Gravity
CMOS	Complementary Metal Oxide Semiconductor
COM	Computer On Module

Stellenbosch University http://scholar.sun.ac.za

CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DCM	Direction Cosine Matrix
DEM	Depth Elevation Map
DF	Direction Finding
DGPS	Differential Global Positioning System
DOF	Degree of Freedom
DP	Douglas Peucker
DSP	Digitial Signal Processing
EID	Extended Identifier
EKF	Extended Kalman Filter
ESL	Electronic Systems Laboratory
FAST	Features from Accelerated Segment Test
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FLANN	Fast Library for Approximate Nearest Neighbours
FOV	Field of View
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input Output
GPU	Graphical Processing Unit
GUI	Graphical User Interface
GX	Gumstix
HIL	Hardware in the Loop
HOG	Histogram of Oriented Gradients
I2C	Inter Integrated Circuite
IO	Input/Output

Stellenbosch University http://scholar.sun.ac.za

IMU	Inertial Measurement Unit
IPP	Integrate Performance Primatives
IR	Infra Red
KNN	K-Nearest Neighbour
LAPACK	Linear Algebra Package
LIDAR	Light Detection and Ranging
LMS	Least Mean Square
LPM	Linear Predictor Model
LQ	Linear Quadratic
LQR	Linear Quadratic Regulator
LMLS	Lattice Mean Least Square
LU	Lower/Upper Triangular Matrix
LT	Locktime
MATLAB	Mathematical Laboratory
MAV	Micro Aerial Vehicle
MCA	Minor Component Analysis
MMSE	Minimum Mean Square Estimate
MPC	Model Predictive Control
MSER	Maximally-Stable Extremal Region Extractor
MSL	Meters above Sea Level
NED	North-East-Down Inertial Coordinates
NN	Neural Networks
NPC	Neuro Predictive Controller
OBC	Onboard Computer
OpenCV	Open Computer Vision Library
ORB	Orientated BRIEF

Stellenbosch University http://scholar.sun.ac.za

PA	Prony Analysis
PC	Personal Computer
PD	Proportional and Differential Control
PID	Proportional, Integral and Differential Control
POI	Point of Interest
POP	Package on Package
PX	Pixels
RAM	Random Access Memory
RANSAC	Random Sampling Concensus
RAST	Recovery Assist, Secure and Transverse
RC	Remote Control
RF	Radio Frequency
RLS	Recursive Least Square
RUAV	Rotary Winged Unmanned Aerial Vehicle
RX	Receive Buffer
SDC	State Dependent Coeffiecient
SDRE	State Dependent Ricatti Equation
SHT	Standard Hough Transform
SIFT	Scale Invarient Feature Transform
SLAM	Simultanious Localization and Mapping
SNR	Signal to Noise Ratio
SPI	Serial Peripheral Interface
SUN	Stellenbosch University
SURF	Speeded Up Robust Feature
SVD	Singular Value Decomposition
TBB	Thread Building Blocks

Stellenbosch University http://scholar.sun.ac.za

TI	Texas Instruments
TX	Transmit Buffer
UART	Universal Asynchronous Receiver/Transmittor
UAV	Unmanned Aerial Vehicle
VCAN	Vision on CAN
VL	Visual Lock
VPAM	Visual Position and Attitude Measurement
ZOH	Zero Order Hold

Stellenbosch University http://scholar.sun.ac.za

Acknowledgements

I would not have been able to successfully complete this project, were it not for the following:

- I would like to thank Doctor Iain Peddle, as my supervisor, in guiding me through this project.
- To Professor Thomas Jones, for facilitating and envisioning the ESL.
- To Armscor, for providing the necessary funding for this research project.
- To Emile Rossouw, Carlo Van Schalkwyk and especially AM de Jager for laying down such a great research foundation for autonomous helicopter flight.
- I would like to thank the two ESL lab engineers: Anton Runhaar and Lionel Basson for their contributions in the development of the VCAN board and their efforts in resurrecting the 3-DOF pneumatic platform.
- To Chris Jaquect, for his helpful insights in native programming and Linux systems.
- To Ruan de Hart, for being such a great debugging soundboard.
- Finally, I would like to thank my fiancé Amore, for all her loving support.

Stellenbosch University http://scholar.sun.ac.za

Chapter 1

Introduction

(a) Bell UH-60 approaching flight deck [3]

(b) AH-1 Cobra awaiting authorization to land [4]

It takes great skill and years of maritime experience to successfully land a helicopter on the flight deck of a ship. The helicopter pad is usually located at the stern of the ship, which translates to large sway and heave motions in high-sea conditions. The final stretch of the flight envelope is the most critical, as the pilot not only needs to anticipate the motion of the ship, but also adapt to the rapidly changing airflow-dynamics as the helicopter approaches the flight deck. Generally the autopilot onboard the helicopter is partially disengaged prior to a landing, this is in order to give full control to the pilot, who has extensive experience and honed instincts in executing a safer landing.

Shipboard landings, in high-sea conditions, can be assisted by a haul-down device known as a "Beartrap", which was pioneered by the Royal Canadian Navy. A similar device, known as the Recovery Assist, Secure and Transverse (RAST), is implemented by the U.S. Navy to assist helicopter landings in such conditions.

Stellenbosch University http://scholar.sun.ac.za

An autopilot augmented with an autonomous landing controller can minimize the landing risks and secure a safe landing. Such an autopilot system would also be able to respond much faster to wind disturbances and could monitor the flight deck to statistically forecast on safe landing opportunities. Since this thesis is of a research nature, this project will be implemented on a small scale, Rotary-wing Unmanned Aerial Vehicle (RUAV) which will attempt to land on a three Degree-Of-Freedom (DOF), pneumatic platform. In this project, an onboard vision-system will lock onto a distinct, geometric pattern fixed upon the platform. This vision system will process the captured image to provide the inertial position and attitude information of the flight deck. Onboard motion prediction will forecast a prediction horizon and monitor safe landing opportunities prior to a controlled descent.

1.1 Internal Research

The academic research environment for this project was facilitated within the Electronic Systems Laboratory (ESL). The ESL is situated at the heart of the Electrical and Electronic Department of the Engineering Faculty of Stellenbosch University (SUN). The ESL is the dedicated Unmanned Aerial Vehicle (UAV) research division of Stellenbosch and continuously develops new avionics and environments to benchmark aerial control systems. There have been numerous internal projects, see Figure 1.1, that made a great contribution as research foundation for this project.

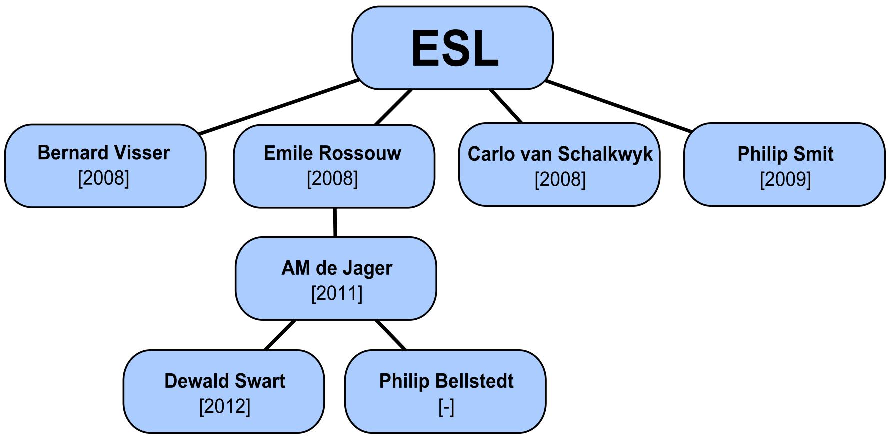


Figure 1.1 – Internal research contributors

Stellenbosch University http://scholar.sun.ac.za

Emile Rossouw [5] successfully developed an autopilot for an unmanned helicopter in 2008. The non-linear model was derived from first principals. The model was decoupled to enable separate controllers and modal analysis was performed on the helicopter's natural modes. Simple, classical successive-loop-closure controllers were designed to stabilize the RUAV. Ground station software was developed to interface with the onboard guidance controller in autonomous waypoint navigation. Practical flight test data supported the functionality of the autopilot.

In 2008 Bernard Visser [6] evaluated possible vision systems to guide an autonomous, fixed-wing UAV onto a landing strip. Light beams, Infra Red (IR) lights and distinct markers were considered as beacons to identify the runway. Both stereo vision onboard the aircraft as well as stereo vision grounded to the runway were considered. A monocular-vision system was proposed and used to target IR markers on the runway, which were validated through Hardware-in-the-Loop (HIL) simulations.

Carlo van Schalkwyk [7] in 2009 converted the existing classical-controllers into Linear Quadratic Regulation (LQR) controllers for the X-Cell Fury Expert unmanned helicopter. Phillip Smit's [8] project originated from the long-term goal to perform an autonomous landing of a unmanned helicopter on a moving deck. A 3-DOF motion simulation platform, capable of simulating the flight deck of a vessel at sea, was developed. Comprehensive controller architecture was developed to regulate the pneumatic actuation of the simulator. Practical motion simulation results of one of the South African Navy Patrol Corvettes demonstrated the success of the platform.

In 2011, AM de Jager [9] augmented the existing kinematic estimator onboard the unmanned X-Cell helicopter with a vision node. Low-cost GPS and inertial measurement units were combined with the high accuracy of a camera system. High-level, guidance algorithms were developed to enable waypoint navigation and autonomous landings. A dedicated Visual Position and Attitude Measurement (VPAM) node was designed and built to execute pose-estimation on IR markers which locate the stationary flight deck. The functionality of the VPAM node was confirmed through a number of practical tests and HIL simulations, but no actual landing was performed.

Stellenbosch University http://scholar.sun.ac.za

In parallel to this thesis, Phillip Bellstedt is in the process of completing a similar thesis of the autonomous landing of a RUAV on a moving flight deck. His project entails instrumenting the flight deck with an avionics pack and relaying flight deck state information to the RUAV. Motion prediction will also be executed off-line and fed to the RUAV. High position accuracy is obtained via a Differential Global Positioning System (DGPS).

1.2 Principal Phases of the Project

This thesis will append on the existing autopilot structure of the X-Cell RUAV and utilize its waypoint navigation outer-loop controller to navigate the RUAV to the last known position of the flight deck using conventional sensors such as GPS, as shown in Figure 1.2, (1). Once the RUAV is within close proximity of the flight deck, it will arm the vision system and lock onto the flight deck identified by the distinct geometric pattern, (2).

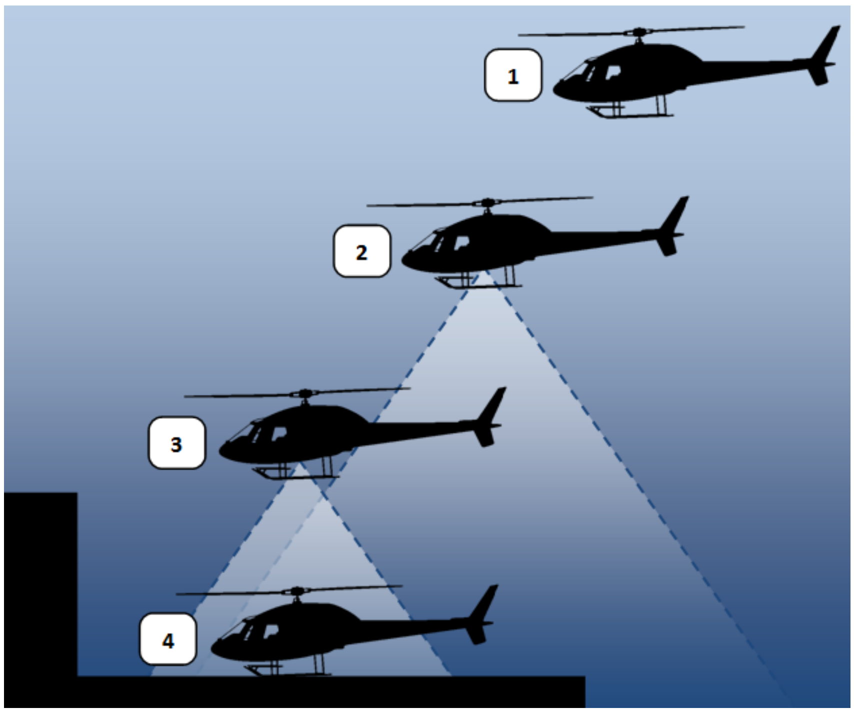


Figure 1.2 – Principal Phases

The RUAV will then latch onto the horizontal position of the flight deck and continue to monitor the deck at a safe hover height, (3). This hover phase is crucial as the RUAV needs to build up a history of the flight deck states (particularly heave, pitch and roll) that will be used to train the motion prediction onboard the RUAV. The predicted future horizon is

Stellenbosch University http://scholar.sun.ac.za

used to forecast safe landing opportunities. Once the RUAV is authorized to land, it will execute a controlled descent which is carefully regulated by the predicted landing time, (4).

1.3 Thesis Layout

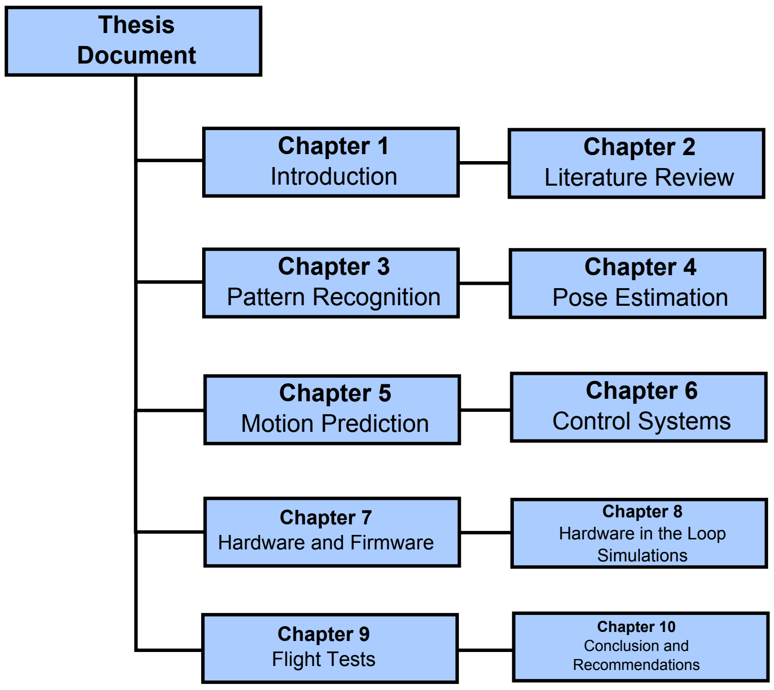

Thesis Document
Chapter 1 Introduction
Chapter 2 Literature Review
Chapter 3 Pattern Recognition
Chapter 4 Pose Estimation
Chapter 5 Motion Prediction
Chapter 6 Control Systems
Chapter 7 Hardware and Firmware
Chapter 8 Hardware in the Loop Simulations
Chapter 9 Flight Tests
Chapter 10 Conclusion and Recommendations

Figure 1.3 – Roadmap of Thesis

The layout of this thesis is in accordance to the roadmap presented in Figure 1.3. A comprehensive literature study of relevant work of computer vision, motion prediction and landing controllers is addressed in Chapter 2. The chosen pattern detection technique used in this project is described in Chapter 3 prior to the presentation of an in-depth, relative pose-estimation formulation in Chapter 4.

Various motion prediction algorithms were weighed against each other in Chapter 5 to select the best predictor candidate for forecasting a safe landing zone. The landing controller and various control aspects are discussed in Chapter 6. Existing as well as new hardware and firmware that were developed by the author will be discussed in detail in Chapter 7.

Stellenbosch University http://scholar.sun.ac.za

Finally, prior to flight tests, the complete system was thoroughly tested in HIL simulations in Chapter 8 to validate the system for flight testing. Flight test data is presented in Chapter 9. This thesis finally concludes and provides project recommendations Chapter 10.

Stellenbosch University http://scholar.sun.ac.za

Chapter 2

Literature Review

The fundamental basis of this project will be the RUAV vision system's ability to lock onto a distinct, geometric pattern fixed onto the flight deck and subsequently extract relative flight deck information from a photo. As such, great emphasis is laid on various pattern detection methods (optical flow, planar object detection, stereo vision and geometric shape skeletons) presented in §2.1.

The RUAV must be able to anticipate safe landing opportunities. As such, various motion prediction algorithms were investigated in §2.2. The onboard predictor will monitor these forecasts and pass them to the landing controller that will regulate a controlled descend.

Using the forecast horizon from the predictor, the RUAV can drive its own states (position, attitude, rates, etc.) to minimize a cost function to track a future reference in a receding prediction horizon. This control technique of earlier compensation is called Model Predictive Control (MPC) and its applicability to this project is investigated in §2.3.

2.1 Computer Vision

Artificial intelligence evolved greatly as the development in computer vision heightened at the turn of the century, lending advance control systems the gift of sight. Computer vision, over the last couple of years, has enjoyed an immense growth as researchers and developers continuously strive to facilitate smarter machine learning algorithms.

Open source Computer Vision (OpenCV) is a freeware C/C++ library of programming functions, developed by Intel, mainly aimed at real-time computer vision and is now supported by Willow Garage and Itseez. It is free for use under the open source Berkeley Software Distribution (BSD) license. The OpenCV library offers a broad range of multi-functionality

Stellenbosch University http://scholar.sun.ac.za

and is cross-platform as it is already ported to the ARM® architecture, a popular Central Processing Unit (CPU) choice for embedded processors.

The greatest asset of OpenCV is the sheer breadth of algorithms included in its standard distribution. It focuses mainly on real-time image processing. The mainline development effort targets the x86 architecture and supports acceleration via the Intel proprietary Integrated Performance Primitives (IPP) library, to accelerate itself using these optimized routines.

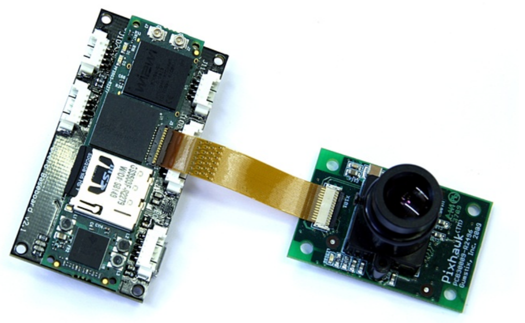


Figure 2.1 – Gumstix Overo COM with Pixhawk Camera [1]

The Gumstix Overo is a tiny but powerful Computer On Module (COM) that performs like a full-sized Linux computer and can be programmed to perform a wide variety of functions in almost any application area. The boards can also expand their peripheral input/output (IO) through a variety of expansion boards. The main features of an Overo COM can include a Texas Instruments OMAP3503 600 MHz processor featuring the ARM® Cortex A8 architecture. The COM features 256MB DDR Random Access Memory (RAM) with 256MB NAND Package-on-Package (POP) memory.

Using a Gumsitx COM, will eliminate the need for such a primitive Digital Signal Processor (DSP) as the dedicated Field Programmable Gate Array (FPGA) used in [9] to pre-process each frame for image markers. The embedded system on the Gumstix COM will be able to execute much more complex image recognition algorithms at a much faster rate, hence the discontinuation on [9]'s project in using IR lamps as flight deck markers. As such, this project is the pioneer in the ESL in using a Gumstix COM to execute onboard vision algorithms. The IR lamps are big and very awkward to mount on the 3-DOF platform that mimics ship motion. The IR lamps are not very modular in design, should this system be implemented in an actual full scale autonomous helicopter landing, it would render the lamps impractical.

Stellenbosch University http://scholar.sun.ac.za

The Gumstix COM supplies ample processing power for a more optimized pattern recognition algorithm and by removing the IR filter on the camera, the camera can serve the dual purpose of surveillance and pattern detection.

In contrast to implementing a pure vision system, [10] developed a novel, mirror-rotating Light Detection and Ranging (LIDAR) sensor to map the relative position and attitude information of a flight deck. This sensor was also complimented by a digital camera to target a light beacon as the initial search bearing for the flight deck and interfaced with a FPGA. The system was augmented to a RMAX helicopter, as seen in Figure 2.2 and succeeded in estimating the inertial states of a pneumatic deck simulator, posing as the flight deck, with great accuracy. No onboard prediction algorithm was documented to forecast a safe landing zone, but a deck-lock system was implemented to secure the RUAV on the flight deck at touchdown.

Similiar to [10]'s LIDAR instrumentation, Ultra-sonic sensors, Doppler and radio triangulation or Direction Finding could be used to provide accurate relative state information and so aid in precision landing.

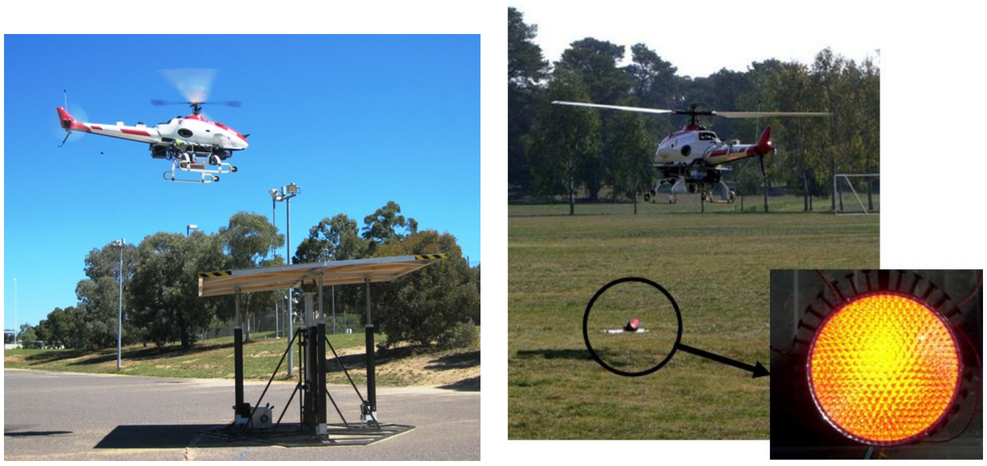

(a) RMAX approaching deck [10] (b) RMAX bearing towards light beacon [10]

Figure 2.2 – RMAX

Stellenbosch University http://scholar.sun.ac.za

2.1.1 Optical Flow

Optical flow is an approximation of the local image motion, based on local derivatives in a given sequence of images. It is assumed that all temporal intensity changes are due to motion only, [11]. [12] used optical flow on a wide angle camera to generate depth-map based collision avoidance and safely navigate a Micro Aerial Vehicle (MAV) through indoor corridors. An omnidirectional fisheye lens was used as the primary sensor, while Inertial Measurement Unit (IMU) data was used to compensate for the rotation based effects of optical flow. The overall computation was done offline.

[13] designed and implemented a vision-based, nonlinear controller to stabilize a quadcopter using optical flow. The image derivatives were fed directly into the simple controllers to stabilize the decoupled, lateral and longitudinal plants of the rotorcraft. [14] implemented onboard optical flow estimation on a Gumstix Overo COM. The rotorcraft's heading estimation was enhanced by computing the phase correlation in the log polar domain of the captured frame. Results are comparable to conventional IMU telemetry.

2.1.2 Planar Object Detection

Planar object detection poses as one of the most challenging obstacles in computer vision. There is a number of variants for planar object detection schemes, but all are very computationally demanding, limiting them from real-time implementations such as object tracking on simple microprocessors. Fortunately, the computation load is optimized by the dedicated efforts of several companies. Examples of such are Intel's Parallel Processing (IPP) or Thread Building Blocks (TBB) which enables multi-core processing on the CPU. A recent OpenCV release also added support for Graphics Processing Unit (GPU) acceleration using the NVIDIA Compute Unified Device Architecture (CUDA) standard.

Planar object detection is based on a feature extractor, descriptor and matcher components. Popular feature extractors are FAST, STAR, Oriented BRIEF (ORB), Maximally-Stable Extremal Region Extractor (MSER). Once these features are enlisted, they are characterized by a feature descriptor such as Binary Robust Independent Elementary Features (BRIEF), Scale Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF) and Boosted Randomized FERNS classifiers. Feature matching is done through Brute force, Fast-Approximate Nearest Neighbor (FLANN) and KNN methods. After a cluster of feature matching pairs are identified, the homography of the projected plane must be computed

Stellenbosch University http://scholar.sun.ac.za

with the Random Sampled Consensus (RANSAC) algorithm to dispose of outliers that might warp the homography matrix solution.

2.1.3 Stereo Vision

The pinhole camera model defines the geometric relationship between a three-dimensional (3D) point and its two-dimensional (2D) corresponding projection on to the image plane. When using a pinhole camera model, this geometric mapping from 3D to 2D is called a perspective projection, [15]. This model can be expanded to stereo vision where feature triangulation is used to reconstruct a depth map.

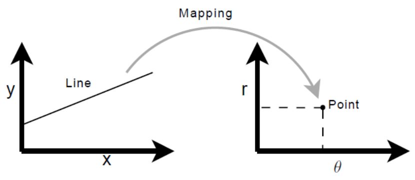
[16] applied stereo vision based relative pose and motion estimation for an RUAV landing. The system was excelled by performing feature point matching on a well-designed landing pattern composed of specific organized polygons. Using stereo vision, the reconstruction of the depth information was computed faster and more precisely than for a single camera. [17] employed stereo vision to construct a depth elevation map of the landing area to identify possible safe landing zones in a hazardous terrain. The flight test data illuminated this project as the first autonomous landing of an unmanned helicopter in an unknown and hazardous terrain.

[18] augmented a MAV with a monocular, Simultaneous Localization and Mapping (SLAM) framework to stabilize the vehicle and overcome both drift and dependency in GPS denied environments. The adapted SLAM algorithm and Proportional and Differential (PD) controllers were executed onboard an Intel Atom computer.

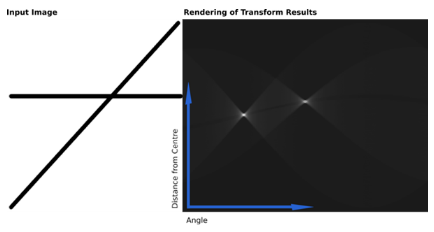
2.1.4 Geometric shape skeletons

The basis of geometric shape skeletons relies on computing translation, scale and rotation invariants in order to distinguish between positive matches of distinct object features and background noise. Such basic methods include block template-matching, shape skeletons, moment invariants, log-polar mapping, geometric shape descriptors, boundary profiles, s-psi plots, Fourier descriptors, active shape models, the Hough transform, the generalized Hough transform and 2D projective transforms. The most popular of these methods are Hough transforms and Hu invariant moments.

Stellenbosch University http://scholar.sun.ac.za



(a) Mapping from Cartesian to Hough space [19]



(b) Hough Domain [19]

Figure 2.3 – Linear Hough Transform

2.1.4.1 Hough Transforms

Initially it was considered to use the classical Hough Transform to identify lines on the flight deck. The intersections of these lines would then form the key features of the landing deck. After the image is captured, all visible lines are deduced from the Canny Edge Filter by applying a Laplace Kernel to the image matrix. The sinusoidal transformation of a single point on the Cartesian plane mapped onto the Hough plane can be seen in Figure 2.3. In the Standard Hough Transform (SHT), each subset point of the visible lines is passed onto an accumulator on the Hough plane. The dominant peaks of the histogram or accumulator is condensed with a maximum-local-suppression algorithm to find the dominant line parameters within the frame. The appeal of this method is its robustness to the partial occlusion of the flight deck pattern. Each line consists of a myriad of points, hence the line and subsequently the key feature-point, remains well represented even if a line is partially concealed. The Circle Hough transform is a variant of the SHT to detect circles within a captured frame.

2.1.4.2 Hu Invariant Moments

A very attractive image contour matching algorithm is the Hu Invariant Moment matching. The nature of these normalized moments makes them invariant to scaling, translation and rotation. The Hu moments are a linear combination of the later central moments. [20] performed helipad detection and flight deck state estimation from an unmanned helicopter, through Hu Invariant Moments, to filter out false candidates of the flight deck. Points of interests are tracked via the Lucas-Kanade optical flow method. The state information is obtained by the iterative minimization of the back projection error of the estimated states.

[21]'s experimental test-bed Autonomous Vehicle Aerial Tracking And Reconnaissance (AVATAR) illustrated the accuracy and repeatability of their vision system through flight trials for a stationary flight deck. Real-time helipad detection was performed where the

Stellenbosch University http://scholar.sun.ac.za

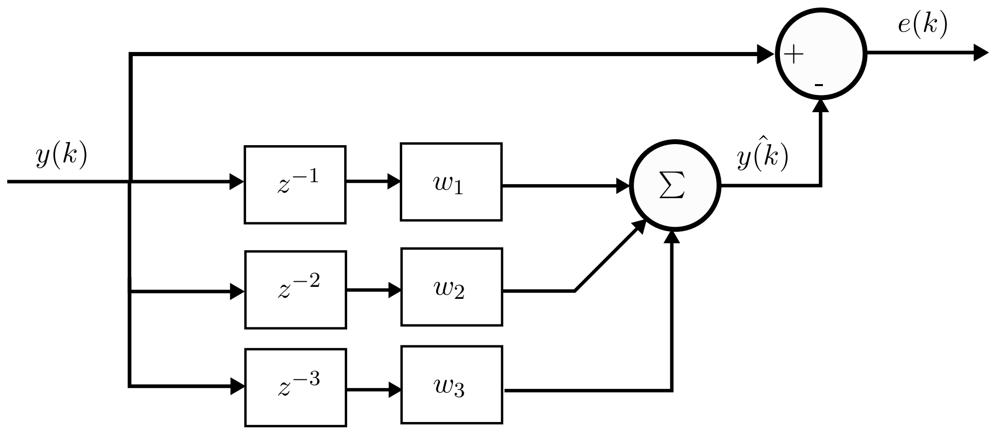


Figure 2.4 – Tap-delay cascade of a FIR filter

captured frame was compared to a pre-recorded, geometric, skeleton data bank of helipads. The latter project was expanded by [22] in the autonomous landing of a helicopter on a translational moving target. The flight deck's linear trajectory was captured by the simple linear dynamics of a Kalman filter and used for intercepting it in a path planner. The flight tests of the UAV are still ongoing.

2.2 Motion Prediction

"History teaches everything, even the future"

Alphonse de Lamartine, French Poet, 1847

It is crucial that the helicopter's path planner can anticipate the motion of the flight deck for a successful landing. The problem in this application is that the helicopter will have no knowledge of the flight deck motion prior to its hover above the flight deck.

It is imperative that the helicopter should hover above the flight deck and the monitor deck to populate a short learning period. The latter learning period is used in the prediction forecast on the flight deck's states and subsequently the RUAV can commit to a safe landing time if conditions permit it. More advanced control techniques, such as MPC, could contribute greatly as this forecast can be injected directly into the landing controller, which is of great benefit.

The basis of Linear Predictor Models (LPM) is based on an all-zero Finite Impulse Response (FIR) filter, as depicted in Figure 2.4, or more commonly known as an Auto Regression

Stellenbosch University http://scholar.sun.ac.za

model (AR). Each future prediction is then calculated step-wise and used to update the next auto regression row vector prior to another prediction step.

$$y(\hat{k}) = \sum_{i=1}^{N} w_i(k)y(k-i) + e(k) \tag{2.2.1}$$

Applications of adaptive filters include: system identification, inverse modeling, linear prediction and interference cancellation. There are numerous methods to train the auto regression weights, w_i in Equation 2.2.1, such as the ordinary least squares procedure, method of moments through the Yule Walker equations, or Markov chain Monte Carlo methods or the Levinson Durbin recursion methods.

[23] developed an adaptive multi-step linear predictor of optimal order based on the Bayes information criterion to predict on the uncertain tendency of ship motion dynamic variations and stochastic sea state disturbances. The proposed predictor outperformed a predefined order predictor and an AR predictor. This predictor however used a very large training window that could be a great draw-back in predicting on the quasi-inertial states of the ship deck should the helicopter hover that long above the deck. A large matching window would include the dynamics of conventional GPS random walk or drift.

[24], [25] outlined the basics of linear prediction systems and drew a comparison between the autocorrelation method and the covariance method as internal dynamics of a linear predictor in a human speech production application. [26] suggested adapting the predictor coefficients by either looking at the Minimum Mean Square Estimate (MMSE) or Weighted Least Square Error (WLSE). The MMSE was performed either by the Levinson-Durbin Recursion or the calculation of the autocorrelation coefficients. WLSE was performed in a recursive least square iteration with an applicable forgetting factor. The WLSE predictor is more applicable in real-life applications since it circumvents the calculation of the autocorrelation coefficients.

Kalman Filter (KF) variants for uncertain noise covariance [27] and auto-regression models fail to maintain a high degree of accuracy in a prediction window longer than 2-3 seconds as they cannot adapt fast enough under noisy circumstances. [28] used artificial neural networks which were trained with Singular Value Decomposition (SVD) and genetic algorithms to predict suitable helicopter and aircraft deployment and recovery windows. It was shown that

Stellenbosch University http://scholar.sun.ac.za

the high and low amplitude wave motion was accurately represented up to a window of 10 seconds. Several signal prediction algorithms were investigated.

2.2.1 Fast Fourier Transforms

Fast Fourier Transforms (FFT), Equation 2.2.2 have been the pioneer in determining the frequency content of any signal. The drawback from using a FFT for signal prediction is that it does not estimate damping characteristics and so can not accurately represent transient signals portrayed by ship motion.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi k \frac{n}{N}} [k = 0, ...N-1] \tag{2.2.2}$$

2.2.2 Prony Analysis

The driving mechanism of a Prony analysis, formulated from [29] is the curve fitting of a set of complex exponentials, Equation 2.2.3, in the time domain on a cluster of data points. Each of these exponentials contains a magnitude, damping factor, phase and frequency component for estimation. Furthermore, since it is able to estimate damping factors, it is ideal to estimate transient signals, where FFTs lack the ability.

$$y(t) = \sum_{n=1}^{N} A_n e^{\sigma_n t} cos(2\pi f_n t + \theta_n) \tag{2.2.3}$$

The regression matrix is generally not square, hence the Moore-Penros pseudo inverse is computed to solve the linear coefficients of the system. Gauss-Jordan elimination, Lower-/Upper Triangular Matrix decomposition, Eigen decomposition and Cholesky decomposition are a few common matrix inversion techniques. The roots of the characteristic polynomial given by the solution for the linear coefficients of the latter system, will portray the damping factors and frequencies respectively.

[29] compared accuracy and computational complexity of the Burg algorithm, the Maple algorithm and the SVD algorithm to solve the coefficients of the LPM. The disadvantage of the Prony method is that it is very computationally expensive as it requires a powerful programming library to handle complex matrix operations like Linear Algebraic Package (LAPACK). Prony is also very intolerable to even the slightest signal noise and order size, causing this method to identify false damping factors greater than unity, rendering the

Stellenbosch University http://scholar.sun.ac.za

system totally unstable in a prediction forecast. The Prony system could be solved in a Recursive Least Square (RLS) algorithm or an immediate SVD.

2.2.3 Recursive Least Square

The RLS adaptive filter is an algorithm which recursively finds the filter coefficients that minimize a weighted linear least squares cost function relating to the input signals. This is in contrast to other algorithms such as the LMS that aim to reduce the mean square error. In the derivation of the RLS, the input signals are considered deterministic, while for the LMS and similar algorithm they are considered stochastic. Compared to most of its competitors, the RLS exhibits extremely fast convergence. However, this benefit comes at the cost of high computational complexity.

The discrete coefficients can be calculated recursively through the KF gains that is augmented with a forgetting factor that regulates the gains' sensitivity to past samples. Note that the recursion for the covariance matrix follows an Algebraic Riccati (AR) equation and thus draws parallels to the KF, [30]. The KF is everywhere renowned as the father of optimal estimation, with many more variations such as extended and unscented filters. The problem however that handicaps the use of this filter is that the trend in the ship motion shifts too quickly for even a recursive implementation of a KF.

2.2.4 Minor Component Analysis

[31] developed a high performance ship-motion prediction algorithm based on Minor Component Analysis (MCA). Simulation results concluded that this method can predict ship motion with consistent accuracy for a long period of time. This method was also compared with other conventional algorithms such as Neural Networks (NN), AR methods and the Wiener prediction. The main advantage of this method is that this method is much less computationally demanding than the Prony method. The non-square auto-regression matrix may not be square and since it suffers from singularity, its pseudo inverse needs to be computed via SVD. SVD is much faster than the Levinson-Durbin recursion methods or Gauss-Newton elimination methods.

MCA utilizes the null space of the auto-regression matrix and equates the LPM coefficients to the Eigen vector corresponding to the smallest singular value of the autoregression matrix. MCA has the unfortunate trend that it estimates discrete poles outside the unity circle in the presence of a noisy input signal. It accurately matches the matching window, but disregards

Stellenbosch University http://scholar.sun.ac.za

the fact the some of the estimated poles will render the predictions unstable if propagated forward into the future horizon.

2.2.5 Lattice Least Mean Square

The Lattice Mean Least Squares (LLMS) adaptive filter is related to the standard RLS except that it requires fewer arithmetic operations (order N). It offers additional advantages over conventional LMS algorithms such as faster convergence rates, modular structure, and insensitivity to variations in Eigen value spread of the input correlation matrix. Noisy measurements are a reality in any helicopter. The aim of LLMS is to minimize a cost function, J, that penalized the matching error across the learning window and is defined by:

$$J(k) = \sum_{i=1}^{M} e^2 = [(B(k) - A(k)X(k))]^T Q[B(k) - A(k)X(k)] \tag{2.2.4}$$

X is the LPM coefficients, A the learning-history Toeplitz matrix and B, the matching window column vector. Q is a diagonal quadratic weighing matrix for the samples within the matching window. X is then trained into the direction of the steepest decent that minimizes the cost function.

$$\frac{dJ}{dX}(k) = -2A(k)^T Q[B(k) - A(k)QX(k)] \tag{2.2.5}$$

$$X(k+1) = X(k) - \mu\frac{dJ}{dX}(k) \tag{2.2.6}$$

μ is the learning rate constant that adheres to the Euler Stability Criterion discussed in Chapter 5. Notice that this update ignores the quadratic and higher order terms of the Taylor series expansion. The weighing coefficients can then be used to propagate a linear predictor.

2.3 Model Predictive Control

The forecast of the flight deck states by the prediction algorithm will complement a MPC controller beautifully since MPC tries to minimize a cost function or future cross-track error on a reference defined for a receding horizon.

[32] and [33] laid emphasis on the existence of stabilizing feedback by solving closed loop full stated feedback gains of a LQR controller via the State Dependent Ricatti Equation (SDRE), [34]. [35] investigated the stability and optimality issues for a MPC controller on

Stellenbosch University http://scholar.sun.ac.za

an autonomous helicopter and suggested first stabilizing the system with initial feedback through a LQR controller with successive linearization. [36] used MPC for attitude control on a Vario unmanned helicopter. Remarkable performance was obtained and it was concluded that MPC is ideal for multivariable systems with none-minimum phase and time delays and input constraints like the servo motors.

[37] evaluated the tracking performance of a helicopter to a moving platform under a controller that solves a SDRE in real-time for stabilizing feedback. To force the controller into the State Dependent Coefficient form, a nonlinear compensator is added. Non-linear simulation results showed satisfactory tracking in the final flight envelope of the helicopter. [38] designed and implemented a MPC for a model toy helicopter with state estimation and recursive modal adaptation. Results concluded that MPC significantly out-performed classical Proportional, Integral and Differential (PID) type controllers for the highly non-linear and unstable dynamics of a helicopter. The computational intensity of the control loop, however, forced it to be executed on a separate dedicated xPC microprocessor.

[39] conducted a theoretical study of their MPC's performance if it is augmented with a Fuzzy Compensator and thus becomes a Neuro-Predictive Controller. It was seen that this new controller greatly increased the system performance in overshoot and settling time. The system also became much more robust to disturbances and parameter changes. [40] illustrated the superiority of a model predictive neural controller over a SDRE or LQ controller in aggressive helicopter flight manoeuvres.

2.4 Closure

Various pattern detection methods were mentioned in §2.1 to locate flight deck pattern image markers within a photo. The proposed pattern recognition algorithm is introduced in Chapter 3. Pose estimation will be performed on the image markers, as discussed in Chapter 4, to provide a measure for the flight deck inertial states. The latter states will be observed and imported into the proposed motion prediction algorithm which is introduced in Chapter 5. The LLMS adaptive filter was chosen to be the best candidate for a LPM.

The forecast of the flight deck states by the prediction algorithm will complement a MPC controller beautifully in a precision landing framework. The original UAV system architecture

Stellenbosch University http://scholar.sun.ac.za

of the ESL relied on a PC104 stack, that enabled more complex control like LQR and modern control like MPC could be implemented with ease. In recent light to minimized power consumption and weight, the avionics of the ESL have been ported to microprocessors which severely limits advance control techniques for this system. As such, MPC will not be implemented. The heave controller will be discussed in Chapter 6.

Stellenbosch University http://scholar.sun.ac.za

Chapter 3

Pattern Recognition

This section describes the pattern detection process of identifying the geometric pattern of the flight deck within a captured frame. This phase is very important since image markers need to be allocated onto the features detected within the frame. These markers are then used to perform pose estimation. A brief overview of the selected pattern detection algorithm will be presented in §3.1, followed by a discussion of various booster techniques to enhance this method in §3.2.

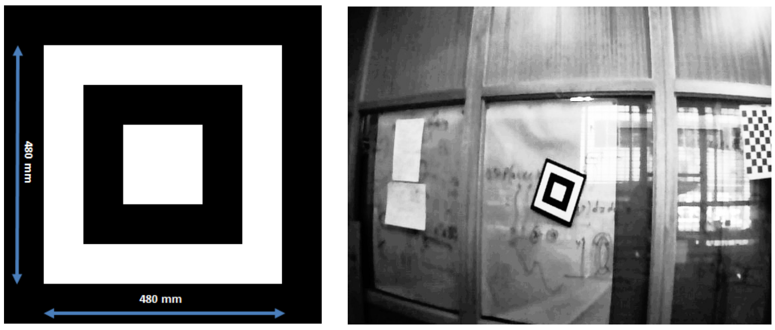
3.1 Method Overview

The basis of this thesis was to develop a vision sensor that would be able to extract the states of a moving flight deck. A very robust image recognition technique that requires minimal computation time is outlined in Figure 3.2. This method relies on definite contrasts within a photo and is an iterative search scheme that passes through a logical linked fern structure which is populated with contours detected within the photo.

Various patterns were considered to identify the flight deck, but ultimately three concentric squares, Figure 3.1, were chosen for the following reasons:

- Squares do not require computing computational demanding Hu invarient moments or Hough Transforms to identify their simplistic shapes.
- Squares are also very distinct shapes whose corners can be easily approximated.
- In partial occlusion of the geometric pattern when the RUAV is in very close proximity to the flight deck, concentricity of the pattern will allow the vision system to still track the flight deck.

Stellenbosch University http://scholar.sun.ac.za



(a) Selected Pattern (b) Pattern in background

Figure 3.1 – Flight Deck Pattern

- A minimum of three markers are needed to estimate a plane, but this pattern is marker dense and provides twelve markers for a better defined solution.

As for the dimension of the geometric flight deck pattern, a pattern size of 48 cm x 48 cm was chosen for the following reasons:

- This pattern size relates to a comfortable virtual hover height in the pre-flight lighting calibration (refer to Chapter 9).
- The smaller pattern will relate to a larger tolerable radial drift away from the flight deck at the hover height and complete Visual Lock (VL) on the pattern will also be lost very, very close to the pattern
- In rotorcraft airflow dynamics, ground effect is experienced roughly within one rotor length of the ground. For the X-Cell Fury Expert RUAV, this translates to a hover height where VL on the largest square will be lost marginally.

The operating system running on the Gumstix COM is Linux based. The Linux video driver of the camera is configured to populate a 8-bit greyscale image matrix (V4L2-PIX-FMT-GREY). This matrix is simply a degenerate of the Y'CbCr format that contains no hue or saturation component (Cb or Cr), but only retains the luminosity data. This image matrix is then appropriately cast to OpenCV's IplImage (IPL-DEPTH-8U). The generated image is first passed through a kernel filter to remove noise segments. The kernel filter in this application equates the filtered pixel value to the weighed neighbouring pixel values within a 7x7 grid according to a Gaussian distribution. This before it is passed through a cut-off filter with an automatic threshold value to generate a purely binary image, *cvThreshold*.

Stellenbosch University http://scholar.sun.ac.za

OpenCV's *cvFindContours* method is called to enlist all the detected contours in the binary frame into a logical linked tree or a fern structure as can be seen in Figure 3.2. There are two types of nodes as shown in Figure 3.2; containers (c0,c000,...) and holes (h00,h01,...). This method, Suzuki85, is based on the topological structural analysis of digitized binary images by border following, [41].

The deck pattern classifier passes down to each leaf of the contour fern. Each leaf has a counter that increments to the previous ancestor, but only if the current contour can be successfully condensed to four dominant points and does not touch the boundary of the frame. The reduction in points is compulsory to eliminate the redundant nodes that contribute no significant information regarding the fundamental shape. There are numerous popular contour condensing methods to deliver the best perceptual representations of the original lines. Some of these include Teh-Chin, Rosenfeld-Johnson and Douglas-Peucker approximation. The latter method is used. This method, [42], iteratively approximates a polygon by adding additional vertices until the shape approximation falls under a form-closeness bound. Form-closeness is the negative area between the original contour and the condensed contour. Once the deck is detected, the corners are sorted appropriately and passed to the pose estimation process. Using such a robust pattern detection method, eliminates the computational demand of a corresponding feature point algorithm such as RANSAC.

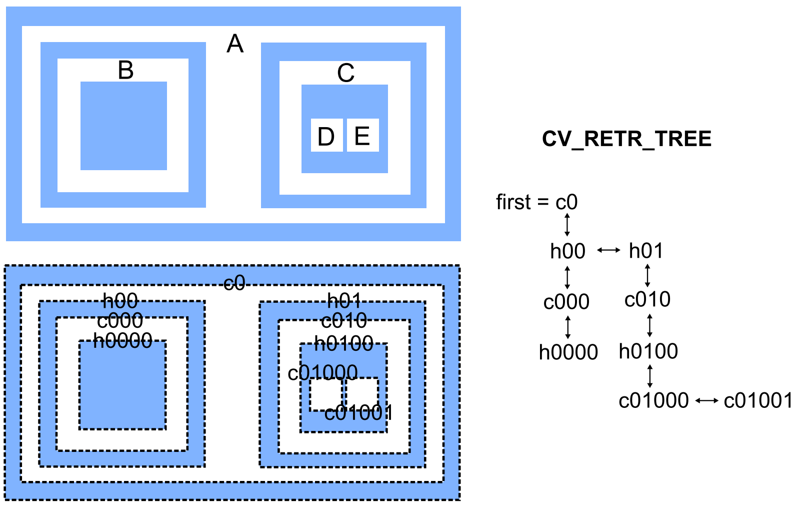
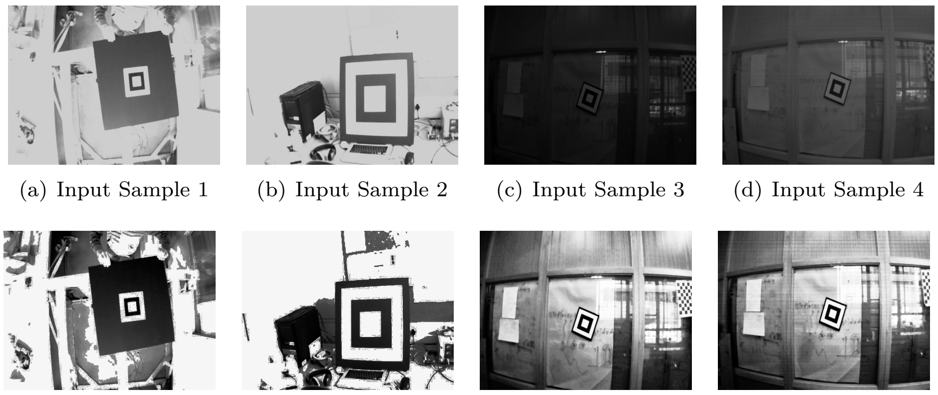


Figure 3.2 – Contour Fern

Stellenbosch University http://scholar.sun.ac.za



(a) Input Sample 1 (b) Input Sample 2 (c) Input Sample 3 (d) Input Sample 4

(e) Ouput Sample 1 (f) Ouput Sample 2 (g) Ouput Sample 3 (h) Ouput Sample 4

Figure 3.3 – Histogram Equalization

3.2 Booster Algorithms

The avionics framework enforces weight and power consumption constraints and so computer vision algorithms are generally slow as they are very computationally demanding. As such the following features were exploited to increase the execution rate of the vision system and make it more robust against different lighting scenarios and partial occlusion of the flight deck pattern.

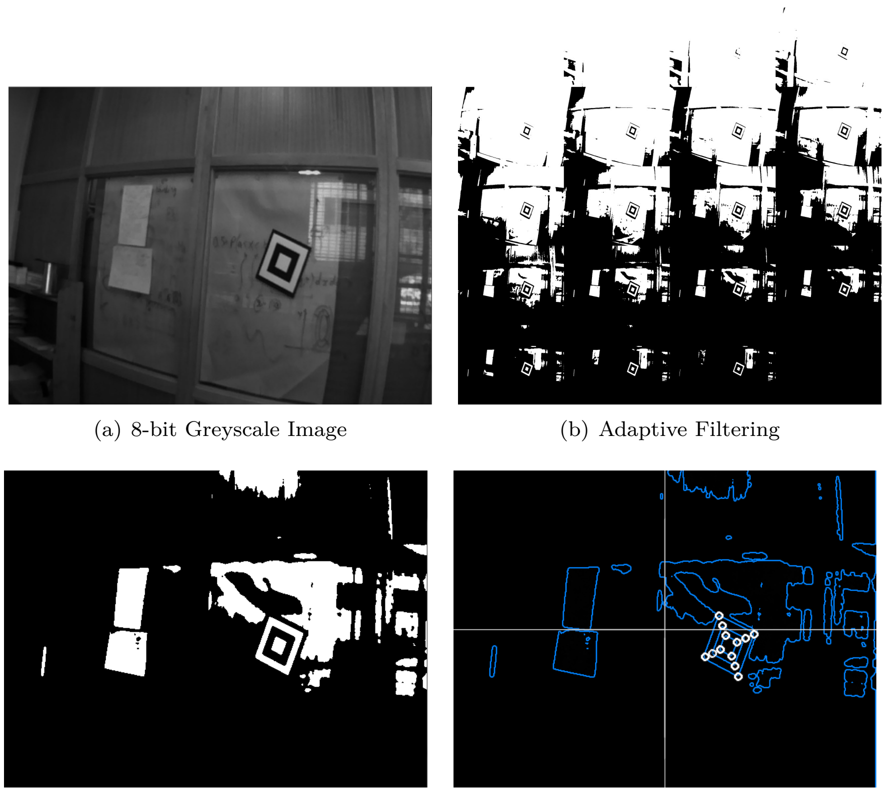
3.2.1 OpenCV DSP Acceleration

OpenCV-DSP-acceleration is an OpenCV ARM-module that is specifically developed to enhance image processing on embedded systems via optimal memory allocation and utilizing instruction sets through compiler optimization and the use of hardware co-processors. Building OpenCV with this module, Chapter 7, enabled the image recognition bandwidth to boost from 10 Hz to 30 Hz.

3.2.2 Contrast and Histogram Equalization

The image's pixels are enhanced with a contrast multiplier. A brightness offset shift is unnecessary due to the fact that the enhanced greyscale image will be passed through a threshold filter in order to reduce it to a black and white image. Furthermore, as a last resort, the image could be pre-processed by a histogram equalizer, *cvEqualizeHist*, to over-emphasize the contours within the frame. These methods are very robust in dramatically increasing the confidence bandwidth of the filter as can be seen in Figure 3.3. The confidence bandwidth of the filter is the percentage of the sightings of the flight deck pattern when the cut-off threshold is iterated from 0 to 255.

Stellenbosch University http://scholar.sun.ac.za


(a) 8-bit Greyscale Image (b) Adaptive Filtering

(c) Optimal Filter (d) Back projection

Figure 3.4 – Deck Detection Pipeline

3.2.3 Automatic Filtering

Edge detection relies on distinct boundaries within a binary image. As such, the threshold filtering makes the pattern detection process very sensitive to lighting. Since the image is not directly streamed down to the ground station, it renders the ground station oblivious to the visible constraints from the camera. Figure 3.4(a) shows the typical greyscale image captured by the Gumstix COM camera. It was further decided to program an automatic filter, which could be initialized from the ground station, which would automatically adjust the threshold optimally to perceive the contours of the flight deck pattern. It entails iterating the threshold value from 0 to 255 and define bands where the deck was detected. The latter process is illustrated in the filmstrip, Figure 3.4(b). The mean of the largest band is then used as the threshold and this bandwidth is then sent down to the ground station to confirm the confidence of the threshold as can be seen in Figure 3.4(c). Figure 3.4(d) shows the re-projection of the estimated states to be validated against the actual corners via the standard deviation of a back projection error discussed in §4.2.

Stellenbosch University http://scholar.sun.ac.za

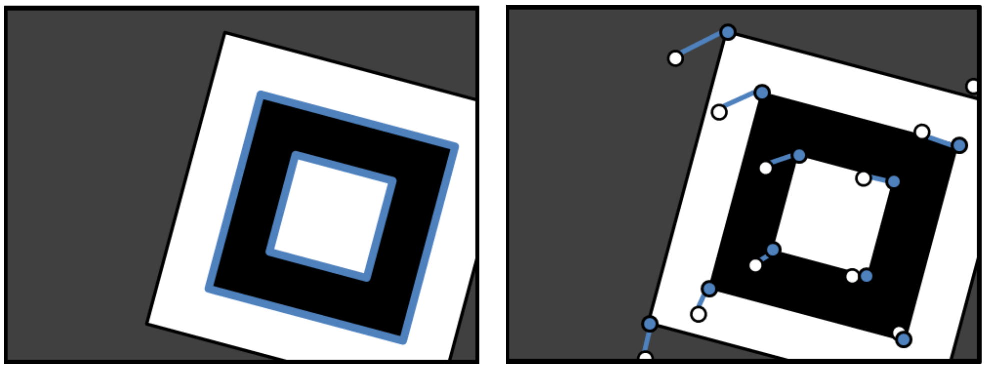

(a) Concentricity (b) Feature Matching

Figure 3.5 – Partial Occlusion

3.2.4 Feature Pairing and Concentricity

The contour tree detection algorithm implies that all three concentric squares representing the flight deck are clearly defined in the frame. There must however, be made provision in the case of partial flight deck pattern occlusion. These scenarios are handled by considering feature pairing and concentricity.

The most recent relative flight deck states are used to re-project the images markers. The dominant points arising from the Douglas-Peucker approximation is used as features. A brute force matcher then pairs the image markers with the detected features based on their nearest neighbour radius. Pose estimation can still be executed even in partial occlusion of the pattern. This method's main drawback is its sensitivity to the neighboring distance threshold. Should VL be temporarily lost and the flight deck states propagate too fast, VL will be lost completely. Should the neighbouring distance threshold be too large, then a image point can latch onto the wrong feature. If this threshold is too small then the system can only tolerate slow relative motion. As such, to simplify matters even further a concentric pattern was chosen.

The beauty of concentricity lies in its simplicity. Numerous projects were conducted were the helipad was stationary and the relative measurements are used to improve the inertial state estimates of the helicopter. Partial occlusion of a set pattern in the case of Hu Invarient Moments, for example, will result in the total loss in VL. By using a concentric pattern, the helicopter can move very close to the deck before VL is lost and so the open-loop descend propagation is postponed until the touchdown.

Stellenbosch University http://scholar.sun.ac.za

3.3 Closure

The contour fern pattern recognition scheme was well formulated in the method overview. The detection scheme was made more robust in different lighting scenarios and various booster algorithms were investigated to optimize the computational load of the pattern recognition pipeline. The detected image markers will now be processed in the pose estimation chapter, Chapter 4, to give extract state information of the flight deck, relative to the RUAV.

Stellenbosch University http://scholar.sun.ac.za

Chapter 4

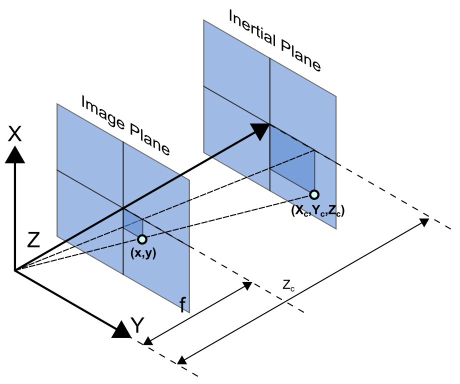
Pose Estimation

After the image markers are assigned to key feature points within the captured frame, as shown in Chapter 3, the relative position and attitude information of the flight deck with respect to the camera needs to be calculated. 3D pose estimation is the problem of determining the transformation of an object in a 2D image which gives the 3D object. The basis of pose estimation is based on the simplicity of the pinhole camera model as described in §4.1. The markers need to be properly described by the pinhole camera model to solve a linear system and extract the relative states through the null space solution as explained in §4.2. The relative states need to be combined by the position and attitude information of the RUAV to provide quasi inertial state estimates of the flight deck. The latter transformations are described in §4.3. Finally, as a preliminary test, lattice noise graphs are constructed during a virtual pose estimation test bed, §4.4, to theoretically quantify the expected noise levels on the extracted relative estimates.

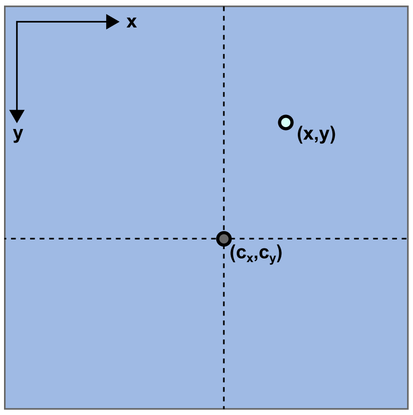
4.1 Pinhole Camera Model

The pinhole camera model describes how a point of interest (POI) is perceived on a focal or image plane. Any POI defined in a coordinate system can be mapped to another by a relative homography matrix and translation offsets as illustrated by Equation 4.1. For the projection of co-planar points, the last column of the homography matrix becomes irrelevant due to the absence of the depth dimension. X_0, Y_0, Z_0 are the coordinates of a POI defined within the flight deck aligned axis. $\bar{X}, \bar{Y}, \bar{Z}$ captures the relative translation information of the flight deck with respect to the camera lens. h_{ji} are the homography matrix elements that captures the relative rotation information of the flight deck with respect to the camera. X_c, Y_c, Z_c is the position of the latter POI coordinated in the camera axis. The convention for

Stellenbosch University http://scholar.sun.ac.za



(a) Pinhole camera model



(b) OpenCV Axis convention

Figure 4.1 – Vision system coordinates

UAV applications within the ESL is the Earth Axes or North-East-Down (NED) coordinate system. The flight range of the RUAV relative to the Earth's dimension, allows one to neglect the Earth's curvature and approximate this geographic coordinate system as a flat inertial coordinate system.

$$\begin{aligned}\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} &= \begin{bmatrix} h_{11} & h_{12} & h_{13} & \bar{X} \\ h_{21} & h_{22} & h_{23} & \bar{Y} \\ h_{31} & h_{32} & h_{33} & \bar{Z} \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} h_{11} & h_{12} & \bar{X} \\ h_{21} & h_{22} & \bar{Y} \\ h_{31} & h_{32} & \bar{Z} \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix}\end{aligned} \tag{4.1.1}$$

The logic behind image projection, x', y', onto a focal plane, as described in Equation 4.1.2, is aspect ratio. The respective focal lengths of the lens is represented by f_x, f_y. The principal point on the image plane, represented by c_x, c_y, is considered to be the centre of the lens, around which distortions occur arising from imperfections.

$$\begin{aligned} x' &= f_x \frac{X_c}{Z_c} \\ y' &= -f_y \frac{Y_c}{Z_c} \end{aligned} \tag{4.1.2}$$

Stellenbosch University http://scholar.sun.ac.za

These imperfections may include the concave nature of a wide-angle lens, resulting in radial distortion. The minor parallel misalignment of the lens with respect to the Complementary Metal Oxide Semiconductor (CMOS) chip results in tangential distortion. These two common phenomena, shown in Figure 4.2, are modelled in Equation 4.1.3 by two radial coefficients (k_1, k_2) and two tangential coefficients (p_1, p_2). If calibrated correctly to describe an accurate model, the distortions bears great benefit as these will vastly increase the Field of View (FOV). The projection coordinate system is given in accordance with OpenCV2.2, which is the open source computer vision library that was used to facilitate the native code architecture of this project.

$$\begin{aligned} x'' &= x'(1 + k_1 r^2 + k_2 r^4) + 2p_1 y' + p_2(r^2 + 2x'^2) \\ y'' &= y'(1 + k_1 r^2 + k_2 r^4) + p_1(r^2 + 2y'^2) + 2p_2 x' \\ r &= x'^2 + y'^2 \end{aligned} \tag{4.1.3}$$

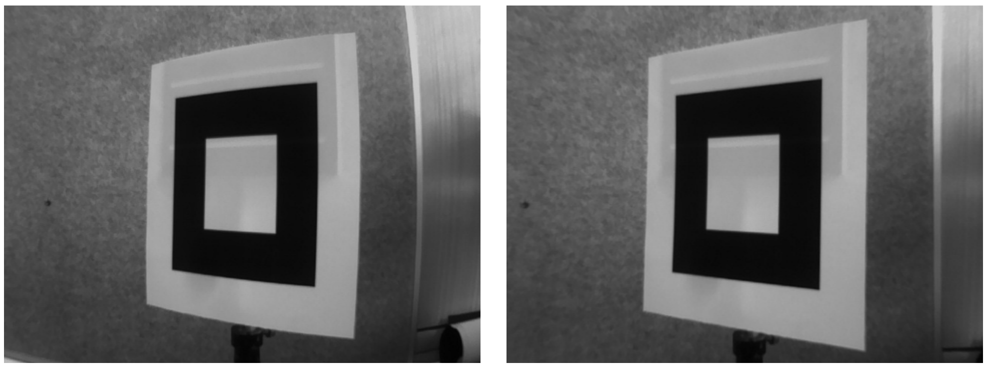

(a) Distorted Image (b) Undistorted Image

Figure 4.2 – Barrel Distortion Compensation

The final image projection point for the POI, modeled to the pinhole camera model, is given by Equation 4.1.4. OpenCV provides an embedded function that estimates the parameters of the fundamental, f_x, f_y, c_x, c_y, and distortion, k_1, k_2, p_1, p_2, parameters. This popular checkerboard calibration is essentially an optimization technique to minimize a global back-projection error on a cluster of sample images. This error is defined by the deviation between a box corner on the actual image plane and that of a re-projection corner described by the fundamental and distortion matrices. The fundamental and distortion parameters are given in Appendix C.

Stellenbosch University http://scholar.sun.ac.za

$$\begin{aligned} x &= f_x x'' + cx \\ y &= f_y y'' + cy \end{aligned} \tag{4.1.4}$$

The pattern detection pipeline populates a cluster of image markers that are matched to their corresponding flight deck pattern corners. Before these markers are injected into the principal matrix, Equation 4.1.5, the markers are passed through a barrel correction algorithm to undistort the markers around their principal point offsets. Since the distortion models are defined only explicitly, the latter compensation is performed as an iterative Gauss-Seidel search method, before the markers are injected into the principal matrix, Equation 4.1.5.

$$\begin{bmatrix} -f_y X_{0_1} & -f_y Y_{0_1} & 0 & 0 & -(y_1 - c_y) X_{0_1} & -(y_1 - c_y) Y_{0_1} & -f_y & 0 & -(y_1 - c_y) \\ 0 & 0 & f_x X_{0_1} & f_x Y_{0_1} & -(x_1 - c_x) X_{0_1} & -(x_1 - c_x) Y_{0_1} & 0 & f_x & -(x_1 - c_x) \\ -f_y X_{0_2} & -f_y Y_{c_2} & 0 & 0 & -(y_2 - c_y) X_{0_2} & -(y_2 - c_y) Y_{0_2} & -f_y & 0 & -(y_2 - c_y) \\ 0 & 0 & f_x X_{0_2} & f_x Y_{0_2} & -(x_2 - c_x) X_{0_2} & -(x_2 - c_x) Y_{0_2} & 0 & f_x & -(x_2 - c_x) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -f_y X_{0_n} & -f_y Y_{0_n} & 0 & 0 & -(y_n - c_y) X_{0_n} & -(y_n - c_y) Y_{0_n} & -f_y & 0 & -(y_n - c_y) \\ 0 & 0 & f_x X_{0_n} & f_x Y_{0_n} & -(x_n - c_x) X_{0_n} & -(x_n - c_x) Y_{0_n} & 0 & f_x & -(x_n - c_x) \end{bmatrix} \cdot \begin{bmatrix} h_{11} \\ h_{12} \\ h_{21} \\ h_{22} \\ h_{31} \\ h_{32} \\ \bar{X} \\ \bar{Y} \\ \bar{Z} \end{bmatrix} = 0 \tag{4.1.5}$$

4.2 The Null Space Solution

[9] conducted an in-depth study on the accuracy and computational intensity between non-iterative minimization techniques, as Levenberg and Marquardt or Gauss Newton methods, to solve this singular system. [9] then resolved to a very fast non-iterative solution; a Singular Value Decomposition (SVD) on the principal matrix and utilize the Eigen vector column that corresponds to the smallest singular value. This solution is more commonly known as the kernel of the matrix, or the null space solution. The much acclaimed OpenCV library subroutines were implemented to perform fast and accurate SVD on the principal matrix. The attitude angles and the translation offsets are only explicitly defined by the null space solution. The null space solution contains six of the nine homography coefficients and three translation offsets, both of which are not scaled appropriately. Fortunately, the normalized, h_{ji} elements are analogues to the elements of a Direction Cosine Matrix (DCM) and must so adhere to their definitions. Since the projection points are all co-planar, the last column of the standard Euler 3-2-1 DCM in Equation 4.2.1 is not displayed as it is not implicitly defined by the null space solution.

Stellenbosch University http://scholar.sun.ac.za

$$\begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{31} & h_{32} \end{bmatrix} = \begin{bmatrix} cos(\psi)cos(\theta) & cos(\psi)sin(\theta)sin(\phi) - sin(\psi)cos(\phi) \\ sin(\psi)cos(\theta) & sin(\psi)sin(\theta)sin(\phi) + cos(\psi)cos(\phi) \\ -sin(\theta) & cos(\theta)sin(\phi) \end{bmatrix} \tag{4.2.1}$$

Utilizing the relative ratios between the h_{ji} elements, the Euler angles can be extracted from the null space solution, followed by the translation offsets as shown in Equation 4.2.2. Λ_i is the i^{th} row element of the null space column and ϕ, θ, ψ represents the relative roll, pitch and yaw angles of the flight deck with respect to the camera. Γ is computed as the scaling factor necessary to correct the null space solution. $\bar{X}, \bar{Y}, \bar{Z}$ represents the relative longitudinal, lateral and vertical offsets of the flight deck with respect to the camera.

$$\begin{aligned} \psi &= -\arctan\left(\frac{\Lambda_3}{\Lambda_1}\right) \\ \theta &= -\arctan\left(\frac{-\Lambda_5}{cos(\psi)\Lambda_1}\right) \\ \Gamma &= \left(\frac{\Lambda_1}{cos(\psi)cos(\theta)}\right) \\ \phi &= -\left(\frac{\Lambda_6}{\Gamma cos(\psi)}\right) \\ \bar{X} &= \frac{\Lambda_6}{\Gamma} \\ \bar{Y} &= \frac{\Lambda_7}{\Gamma} \\ \bar{Z} &= \frac{\Lambda_8}{\Gamma} \end{aligned} \tag{4.2.2}$$

The back projection error, Equation 4.2.3 can be minimized by adapting each of these initial state estimates using the steepest descent from the partial derivative of the back projection error with respect to these parameters. x_i, y_i are the true i^{th} image marker coordinates on the image, $\hat{x_i}, \hat{y_i}$ are the back projection i^{th} coordinates of the proposed solution and N is the number of image markers considered. The computational effort, however, does not justify the negligible increase in accuracy to further adjust these states from their initial estimates. The standard deviation of this back projection error is the criteria to accept the proposed solution.

$$\sigma_{backprojecterror} = \sqrt{\frac{1}{N}\sum_{i=1}^{12}(\hat{x_i} - x_i)^2 + (\hat{y_i} - y_i)^2} \tag{4.2.3}$$

If the proposed solution is not discarded, the latter relative measurements are then combined with the RUAV state estimates from the Extended Kalman Filter (EKF) estimator onboard

Stellenbosch University http://scholar.sun.ac.za

the OBC to provide an estimate of the flight deck states in inertial space. The geometric ambiguity of this simple pattern on the flight deck would not allow the null space solution to estimate the quadrant of the heading of the flight deck. This would not pose as a problem as the RUAV will align to a predefined landing heading.

4.3 Absolute Inertial Pose

[9] utilized the relative pose states of a stationary flight deck to augment the EKF estimator with vision measurements. These measurements were at a much higher frequency and more accurate than single-point *µBlox* GPS. The inertial state estimates of the RUAV were greatly improved once VL was obtained on the flight deck. This project, however, has added complexity of mimicking ship motion and so does not enjoy the luxury of enhancing its own inertial states from the vision measurements. The relative states must be combined with the RUAV onboard estimator to provide an inertial estimate of the flight deck states. To project points that are defined in a set coordinate system to a separate coordinate system, the Euler 3-2-1 DCM transformation is used, as described by Equation 4.3.1. ϕ, θ, ψ are the roll, pitch and yaw angles respectively.

$$DCM_{\psi,\theta,\phi} = \begin{bmatrix} cos(\psi)cos(\theta) & sin(\psi)cos(\theta) & -sin(\theta) \\ cos(\psi)sin(\theta)sin(\phi) - sin(\psi)cos(\phi) & sin(\psi)sin(\theta)sin(\phi) + cos(\psi)cos(\phi) & cos(\theta)sin(\phi) \\ cos(\psi)sin(\theta)cos(\phi) + sin(\psi)sin(\phi) & sin(\psi)sin(\theta)cos(\phi) - cos(\psi)sin(\phi) & cos(\theta)cos(\phi) \end{bmatrix} \quad (4.3.1)$$

Before transforming the relative flight deck states as perceived from the camera into the quasi inertial space, the following definitions are made. $\vec{REL}$ is the vector from the point of origin on the camera lens body axis to the centre of the flight deck. $\vec{CAM}$ is the vector from point of origin on the RUAV body axis to centre of the camera lens. $\vec{HELI}$ is the vector from the point of origin on the quasi inertial axis to the centre of the RUAV. $\vec{DECK}$ is the vector from the point of origin on the quasi inertial axis to the centre of the flight deck. ψ_R, θ_R, ϕ_R are the respective transformation Euler angles of flight deck aligned to the camera lens body axis. ψ_C, θ_C, ϕ_C are the respective transformation Euler angles of camera aligned to the RUAV body axis. ψ_H, θ_H, ϕ_H are the respective transformation Euler angles of RUAV aligned to the axis quasi inertial body axis. ψ_D, θ_D, ϕ_D are the respective transformation Euler angles of flight deck aligned to the axis quasi inertial body axis. The camera's orientation and offsets relative to the centre of gravity of the RUAV and the DGPS Novatel Receiver must be taken into account. These offsets are shown in Figure 4.3, but discussed in Appendix C. Note that the camera lens is labeled as VCAN.

Stellenbosch University http://scholar.sun.ac.za

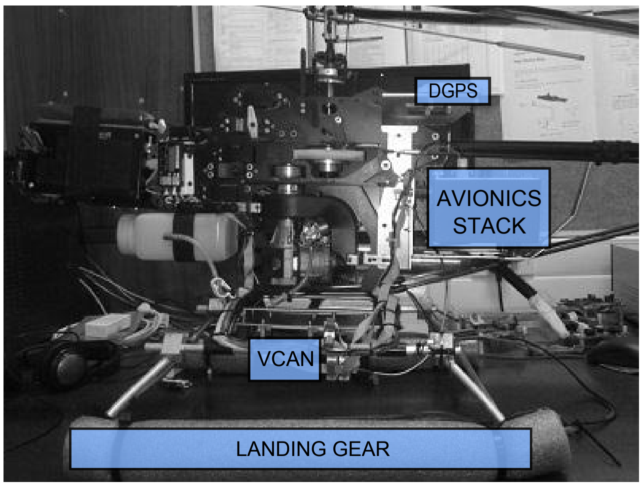


Figure 4.3 – VCAN Positioning

The relative flight deck position defined in the camera coordinated system can be transformed to the quasi inertial axis system by Equation 4.3.2.

$$DE\vec{C}K = (DCM_{\psi_H,\theta_H,\phi_H})((DCM_{\psi_C,\theta_C,\phi_C})RE\vec{L} + CA\vec{M}) + HE\vec{L}I \tag{4.3.2}$$

The relative flight deck attitude defined in the camera coordinated system can be transformed to the quasi inertial axis system by the DCM cascade defined in Equation 4.3.3.

$$DCM_{\psi_D,\theta_D,\phi_D} = (DCM_{\psi_H,\theta_H,\phi_H})(DCM_{\psi_C,\theta_C,\phi_C})(DCM_{\psi_R,\theta_R,\phi_R}) \tag{4.3.3}$$

DCM_{ji} is the j^{th} row, i^{th} column of the DCM matrix. From the latter DCM matrix, the respective transformation Euler angles can be obtained in Equation 4.3.4.

$$\begin{aligned} \phi_D &= \arctan\left(\frac{DCM_{12}}{DCM_{11}}\right) \\ \theta_D &= \arctan\left(\frac{DCM_{23}}{DCM_{33}}\right) \\ \psi_D &= -\arcsin\left(DCM_{13}\right) \end{aligned} \tag{4.3.4}$$

If the telemetry timing between the RUAV EKF estimator states and the vision system is even slightly out of phase, the flight deck estimates will be disturbed by phantom dynamics, which is provoked during aggressive flight manoeuvres. This timing issue is later addressed again in §6.3. The reverse of this pose-estimation process is applied when a virtual image must be generated to test this vision system in simulation. The translation and attitude states of the deck as perceived by the camera must be derived from the true inertial states

Stellenbosch University http://scholar.sun.ac.za

of the flight deck and the RUAV. These relative states can then be used in conjunction with the intrinsic and distortion matrices of the camera to generate a projection of points on the image plane.

The relative flight deck position defined in the camera coordinated system can be derived from the quasi inertial axis system by Equation 4.3.5.

$$\vec{REL} = (DCM_{\psi_C,\theta_C,\phi_C})^T((DCM_{\psi_H,\theta_H,\phi_H})^T(\vec{DECK} - \vec{HELI}) - \vec{CAM}) \tag{4.3.5}$$

The relative flight deck attitude defined in the camera coordinated system can be derived from the quasi inertial axis system by the DCM cascade defined in Equation 4.3.6.

$$DCM_{\psi_R,\theta_R,\phi_R} = (DCM_{\psi_C,\theta_C,\phi_C})^T(DCM_{\psi_H,\theta_H,\phi_H})^T DCM_{\psi_D,\theta_D,\phi_D} \tag{4.3.6}$$

From the latter DCM,the respective transformation Euler angles can be obtained in Equation 4.3.7.

$$\begin{aligned} \phi_R &= \arctan\left(\frac{DCM_{12}}{DCM_{11}}\right) \\ \theta_R &= \arctan\left(\frac{DCM_{23}}{DCM_{33}}\right) \\ \psi_R &= -\arcsin\left(DCM_{13}\right) \end{aligned} \tag{4.3.7}$$

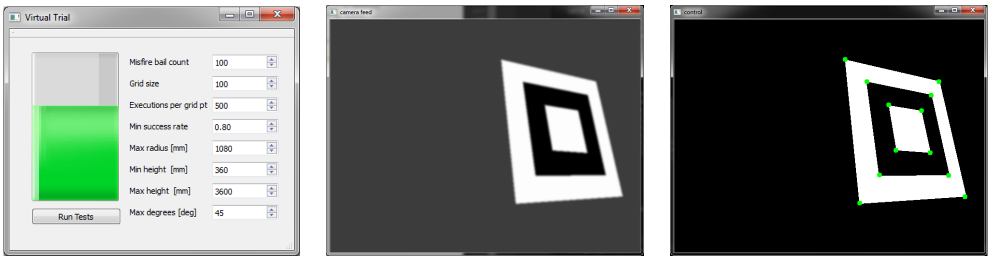
The NovAtel DGPS system will be used as opposed to single-point $\mu Blox$ GPS system in an attempt to isolate the measurement noise to the vision system and validate its performance during flight. The Novatel RT-5 DGPS system provides positional accuracy of 5 cm CEP(Circular Error Probable, i.e. the radius of a circle, centred at the true location of a receiver antenna, that contains 50 percent of the individual position measurements made using a particular navigational system).

4.4 Pose Estimation Testbed

Prior to real camera pose estimation tests, a statistical test bed was programmed to quantify the virtual sensor's theoretical noise levels for relative state estimation of the flight deck. By doing such a theoretical exercise, Figure 4.4, the performance of the vision system can be benchmarked and its noise levels quantified. In this particular study, 5 million photos were generated, making this study impractical to implement in real-life. Another advantage

Stellenbosch University http://scholar.sun.ac.za

of this virtual test bed is that the reference flight deck states are definite, which would be difficult to measure accurately in reality. The fundamental and distortion matrices that were deduced from the actual checkerboard calibration, Table C.1, was used to generate these virtual images. The noise levels from the actual deck tests, Chapter 9, are used to validate the test bed's results. For each state, a lattice graph was constructed with the reprojected state standard deviation error, σ. The base lengths for evaluation were the hover height ratio and drift radius ratio. The hover height ratio is the hover height of the helicopter above the flight deck divided by the side dimension of the largest square. The drift radius ratio is the horizontal radial distance between the helicopter and the flight deck divided by the side dimenstion of the largest square.



Misfire bail count	100
Grid size	100
Executions per grid pt	500
Min success rate	0.80
Max radius [mm]	1080
Min height [mm]	360
Max height [mm]	3600
Max degrees [deg]	45

(a) Statistical analyzer GUI (b) Virtual image (c) Reprojection of relative states

Figure 4.4 – Virtual noise estimation

Figures 4.5, 4.6, 4.7, 4.8, 4.9 and 4.10 display the generated lattice graphs. The hover height ratio ranges from 1-10, the drift radius ratio ranges from 0-5 and the mesh grid is 100 x 100. The reason for such a fine mesh is to increase the resolution of the lattic graph. The maximum roll,pitch and yaw angle deviation were bounded within ± 45 degrees. 500 samples were generated for evaluation at each of the 10 000 grid points. The null regions on the mesh were not populated due to loss in VL. Notice that, as one would expect, a higher drift radius ratio can be tolerated as the hover height ratio increases.

To initially lock onto the flight deck and tolerate a possible drift radius, a high hover height must be chosen. On the other hand, the RUAV has to hover close enough to the flight deck to obtain a tolerable accuracy for the inertial flight deck states for the onboard motion prediction learning phase. The optimal operating point was chosen at a hover height ratio of 3 and the theoretical sensor noise levels that corresponds to this node is given in Table 4.1.

The heave/down, roll and pitch modes of the flight deck are the most critical to monitor for a safe touchdown. Notice that the longitudinal and lateral σ position errors are much

Stellenbosch University http://scholar.sun.ac.za

σ_{north}	σ_{east}	σ_{down}	σ_{roll}	σ_{pitch}	σ_{yaw}
0.0911 mm	0.1750 mm	23.0822 mm	1.1969 deg	1.4152 deg	0.1093 deg

Table 4.1 – State Estimation Noise Levels

smaller than the vertical σ position error. Similarly, the roll and pitch σ angle errors are much larger than the yaw σ angle error. This is due to the fact that the heave, pitch and roll modes are more ill-defined than the other states in the principal matrix and are more difficult to be perceived accurately by a vision system.

Stellenbosch University http://scholar.sun.ac.za

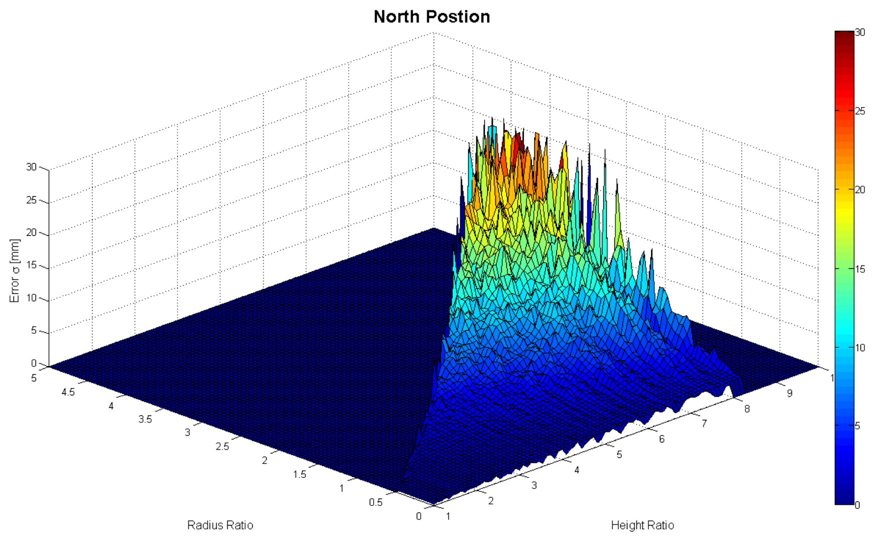


Figure 4.5 – North Error

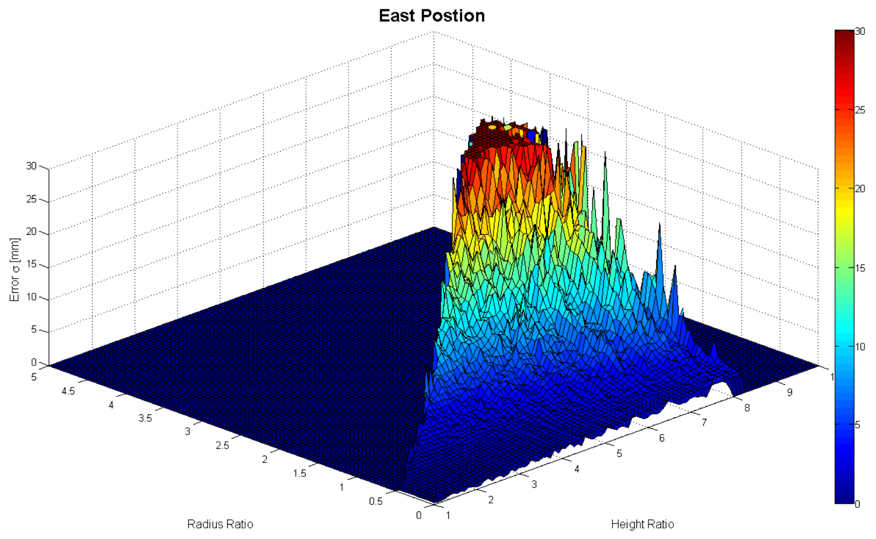


Figure 4.6 – East Error

Stellenbosch University http://scholar.sun.ac.za

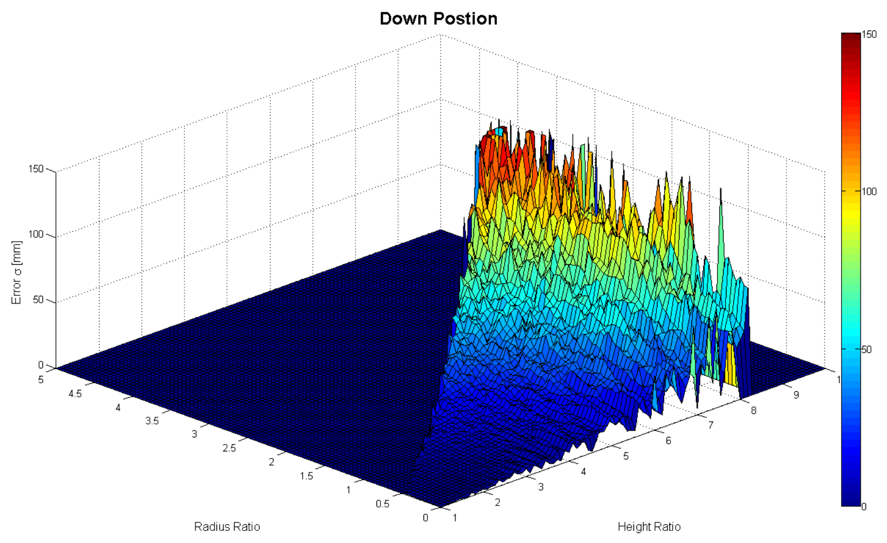


Figure 4.7 – Down Error

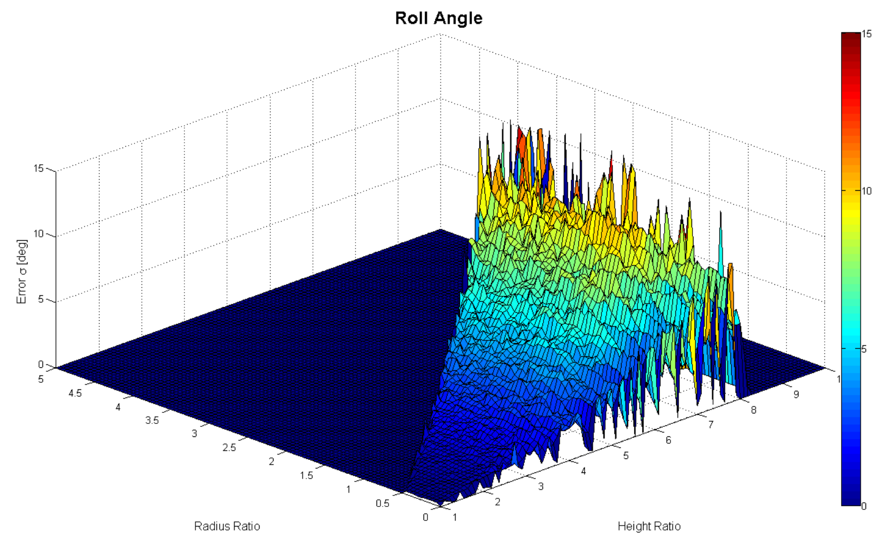


Figure 4.8 – Roll Error

Stellenbosch University http://scholar.sun.ac.za

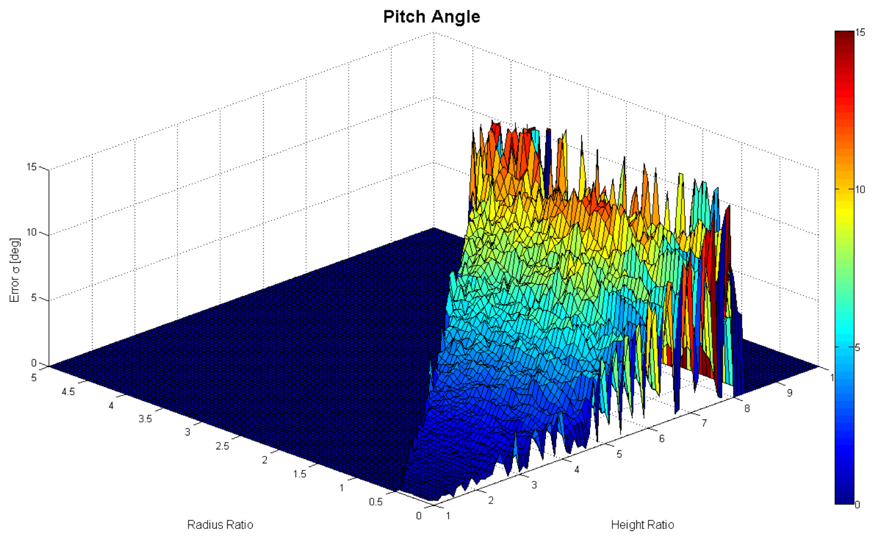


Figure 4.9 – Pitch Error

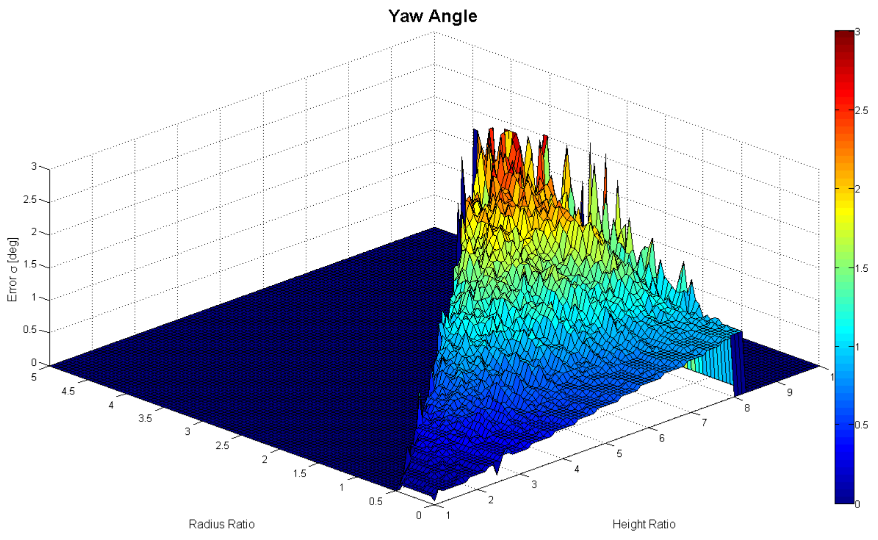


Figure 4.10 – Yaw Error

Stellenbosch University http://scholar.sun.ac.za

4.5 Closure

The image markers detected in Chapter 3 were described by the pinhole camera model. These descriptors were then used to populate a singular principal matrix and its null space solution was utilized to extract the relative states. The latter states are then combined with the states of the onboard kinematic estimator of the RUAV to provide an inertial estimate of the flight deck in inertial space. Before actual tests were conducted, the expected sensor noise levels were evaluated in a theoretical test bed. Now that the flight deck states were well represented, they need to be injected into the motion prediction algorithm to forecast on possible safe landing opportunities in Chapter 5.

Stellenbosch University http://scholar.sun.ac.za

Chapter 5

Motion Prediction

Timing is the fundamental pillar for a RUAV to successfully land on a moving platform. A regression history of the heave, roll and pitch modes of the flight deck, formulated in Chapter 4, will be used to train the motion prediction algorithm. The prediction horizon cast by the latter algorithm will be monitored for suitable landing opportunities.

The fast convergence of the LPM parameters, robustness against input signal noise and minimal computational complexity made the LLMS method, discussed in §2.2, as the best candidate to update the LPM parameters of the motion prediction algorithm. The attributes of this LPM predictor as well as the stability constraints for this method will be discussed in §5.1. The confidence propagation for the forecasting horizon will be discussed in §5.2 prior to the identification of safe landing zones within a prediction horizon, §5.3.

5.1 Stability Criterion

$$J(k) = \sum_{i=1}^{M} e^2 = [(B(k) - A(k)X(k))]^T Q[B(k) - A(k)X(k)] \tag{5.1.1}$$

$$\frac{dJ}{dX}(k) = -2A(k)^T Q[B(k) - A(k)QX(k)] \tag{5.1.2}$$

The latter equations that origin from Chapter 2 §2.2 are displayed for the reader's convience. X is the LPM coefficients, A the learning-history Toeplitz matrix and B, the matching window column vector. Q is a diagonal quadratic weighing matrix for the samples within the matching window. X is then trained into the direction of the steepest decent that minimizes the cost function.

Stellenbosch University http://scholar.sun.ac.za

$$\begin{aligned} X(k+1) &= X(k) - \mu \frac{dJ}{dX}(k) \\ &= \left(I - 2\mu A^T A\right) X(k) + 2\mu A^T B \end{aligned} \tag{5.1.3}$$

From Equation 5.1.3, it is evident that the training algorithm to update LPM parameters of the LRLS method is comparable to discrete explicit Euler step-wise integration. If all back projection errors are weighed equally, then the quadratic weighing matrix, Q, resembles an identity matrix and is omitted from the calculations.

For a fast decaying exponential solution, the convergence of the all LPM parameters needs to be enforced. The marginally-to-stable learning rate, μ must adhere to the Euler Stability Criterion, [30], as described by Equation 5.1.4, where λ_{max} is the largest Eigen value corresponding to the autocorrelation matrix, $R = E\left\{A^T A\right\}$.

$$0 < \mu < \frac{2}{\lambda_{max}} \tag{5.1.4}$$

The rational of this bound is illustrated by the following simple differential equation, $y(t) = y_0 e^{-\lambda t}$, where λ is a system parameter analogues to an Eigen-value. If the derivative is used and the whole system discretized, the Euler integration step is given by Equation 5.1.5.

$$\begin{aligned} \dot{y}(t) &= -\lambda y(t) \\ \frac{y(k+1) - y(k)}{\Delta T} &= -\lambda y(k) \\ y(k+1) &= (1 - \lambda \Delta T) y(k) \\ y(k+1) &= (1 - \lambda \Delta T)^k y_0 \end{aligned} \tag{5.1.5}$$

The solution will only converge if step size adheres to Equation 5.1.6. Since the LPM parameters are updated iteratively per sample instant, the ΔT is analogues to the learning rate μ. Fixed point Newton integration can be applied to update the LPM coefficients numerous runs per sample instant until an error convergence ratio is met.

$$\begin{aligned} |1 - \lambda \Delta T| &< 1 \\ 0 < \Delta T &< \frac{2}{\lambda} \end{aligned} \tag{5.1.6}$$

The graphs that follow illustrate the dynamics of this LLMS prediction algorithm on the inertial heave position of the flight deck. An FFT was performed on the heave motion

Stellenbosch University http://scholar.sun.ac.za

recorded on a frigate during high-sea conditions. The dominant frequency peaks that arose from the latter FFT is used as an input signal to Figures 5.1 and 5.2 and consists of three sinusoidal waves oscillating at 0.22 Hz, 0.17 Hz and 0.13 Hz respectively. The total learning window was 40 seconds with a matching window of 20 seconds at a sampling rate at 10 Hz. Immediately after the predictor was initialized as the matching window of 40 seconds were fully populated, the standard deviation of the back projection error on the matching window, Equation 5.2.1, converges to zero. As an example, forecast monitors were set up at 7.5 and 15 seconds into the future that allows a direct comparision between an earlier and later prediction. The dominant period of the FFT analysis was 5 seconds and as such that would be the minimum prediction horizon to monitor for safe landing oppertunaties. The smooth forecast of the future in Figure 5.2 with a noisy input with a Signal to Noise Ratio (SNR) of 4 facilitates this predictor as an adaptive FIR filter.

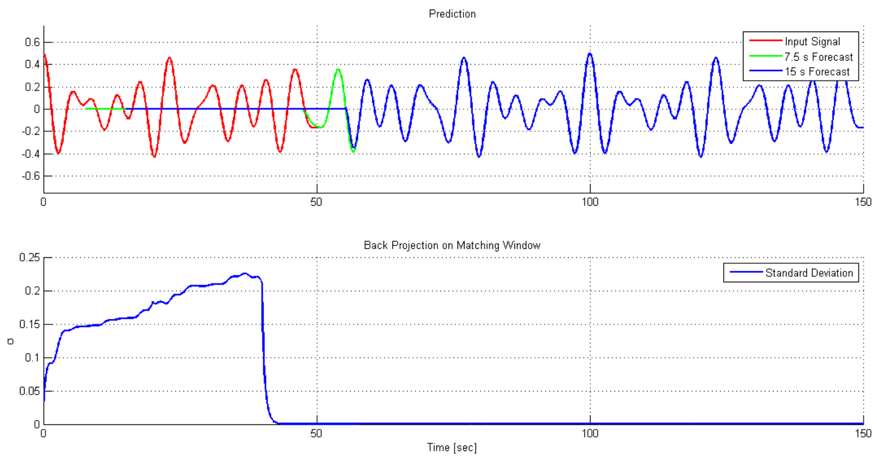


Figure 5.1 – Sinusoidal Motion Predict (No Noise)

The input signal to Figures 5.3 and 5.4 is scaled down, pre-recorded Frigate heave data. Too small learning rate, μ, will inhibit the predictor to train fast enough, yet too large learning rate will ring the predictor unstable as seen in Figure 5.3. To account for parameter variation in the LPM coefficients, the learning rate was safely chosen as 50 percent of the recommended upper limit of the μ constant and will reset automatically once the system is rendered unstable. Forecast monitors were setup at 5 and 10 seconds into the future respectively. From Figure 5.4 it is important to note that the prediction algorithm performs much better during dominant trends (380-480 seconds) as apposed to the smaller trends

Stellenbosch University http://scholar.sun.ac.za

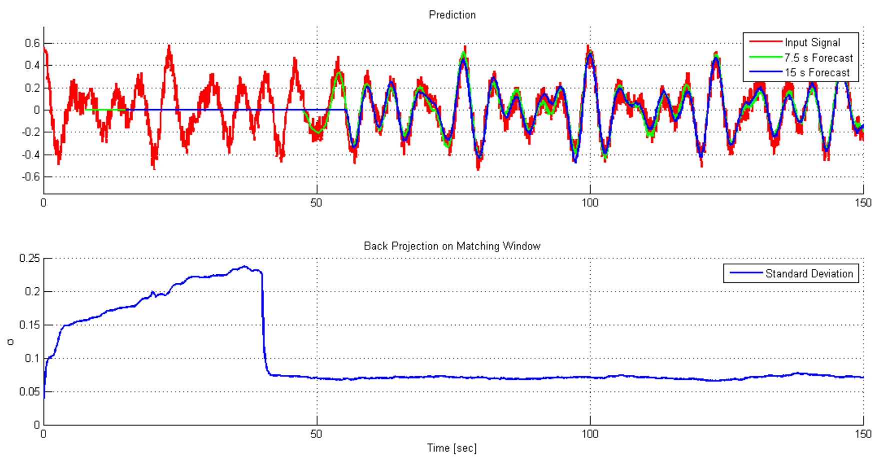


Figure 5.2 – Sinusoidal Motion Predict (SNR 4)

that follows. It should be noted that although the prediction algorithm would not be able to accurately forecast on the magnitude of the heave motion, it captures the phase of the heave motion well. This should however not be a problem since the helicopter's bandwidth is larger than that of the the flight deck and only accurate phase knowledge is required to anticipate the motion of the flight deck.

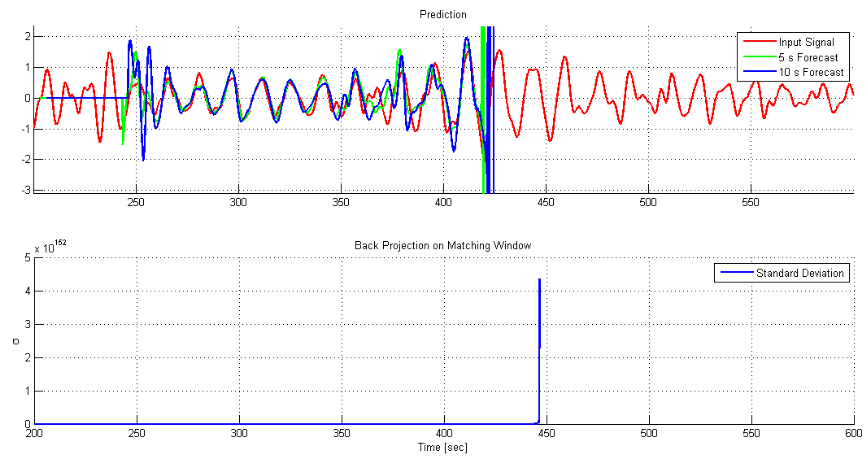


Figure 5.3 – Real Motion Predict (Unstable)

Stellenbosch University http://scholar.sun.ac.za

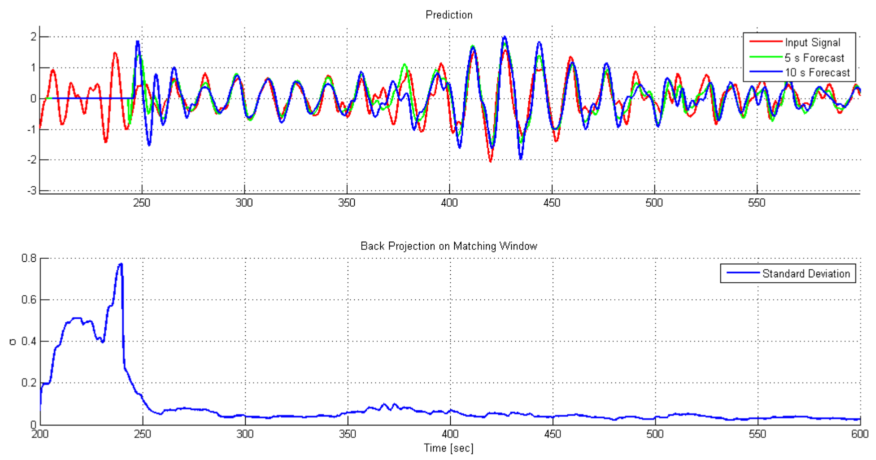


Figure 5.4 – Real Motion Predict (Stable)

5.2 Confidence Intervals

A prediction monitor will survey the heave, pitch and roll of the deck, identifying overlapping viable landing opportunities that adhere to some predefined landing criteria. The furthest safe landing time will be locked unto by the predictor. The latter time or Lock Time (LT) will be sent to the landing state machine. This state machine will regulate the matching offset height from the deck according to the approaching LT. This however makes the ground station oblivious to the justification behind the current LT that will be calculated by the Gumstix COM. So, instead of streaming down the entire prediction horizon, the standard deviation error between the measured and projected deck state in the learning window, will be sent down to the ground station for monitoring, Equation 5.2.1). This standard deviation error serves as a confidence measure in the on-board prediction algorithm for the ground station operator. M is the sample size of the matching window, $y_i(k)$ corresponds to the i^{th} row of the matching measurement column, $B(k)$ and $y_i\hat{(k)}$ corresponds to the i^{th} row of the matching projected column, $A(k)X(k)$. If VL is lost close when to the deck, the prediction would inject the last inertial state of the deck, but this will induce zero-order-hold (ZOH) flat regions in the matching window and so corrupt the trend of the signal.

The proposed solution is to inject the first prediction sample when VL is lost for a measurement. This assures that the learning error at that point will not contribute to the learning adaptation of the LPM parameter, $X(k)$. The helicopter is now able to come very close to the deck and partially lose the deck from the camera, but still predict on the general motion of the flight

Stellenbosch University http://scholar.sun.ac.za

deck.

$$\sigma_{error}(k) = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(y_i(k) - y_i\hat{(k)})^2} \tag{5.2.1}$$

The latter standard deviation can be used to define a 95 percentage confidence interval, CI, for the matching period. This provides an indication of how well the matching performed.

$$CI = \pm 1.96\frac{\sigma_{error}}{\sqrt{M}} \tag{5.2.2}$$

There is, however, a minor drawback to using this prediction method. The uncertainty in the predictor coefficients is not modelled. The implications of this can be illustrated by the following 3-degree predictor.

$$y(k+1) = \theta_1 y(k) + \theta_2 y(k-1) + \theta_3 y(k-2) + e(k+1) \tag{5.2.3a}$$

$$y(k+2) = \theta_1 y(k+1) + \theta_2 y(k) + \theta_3 y(k-1) + e(k+2) \tag{5.2.3b}$$

$$y(k+3) = \theta_1 y(k+2) + \theta_2 y(k+1) + \theta_3 y(k) + e(k+3) \tag{5.2.3c}$$

$$E[e(k)] = E[e(k+1)] = E[e(k+2)] = E[e(k+3)] = \sigma_{error}^2 \tag{5.2.3d}$$

If it is assumed that additional error distribution for all these samples are equal to the error distribution of the matching window, then the standard deviation for the forecast window decays in the binominal pattern shown by Equation 5.2.4

$$\sigma_{k+1} = \sigma_{error}\sqrt{\theta_1^2 + 1} \tag{5.2.4a}$$

$$\sigma_{k+2} = \sigma_{error}\sqrt{\theta_1^4 + \theta_1^2 + \theta_2^2 + 1} \tag{5.2.4b}$$

$$\sigma_{k+3} = \sigma_{error}\sqrt{\theta_1^6 + \theta_1^4 + \theta_1^2 + \theta_2^2 + \theta_3^2 + 2\theta_1^2\theta_2^2 + 1} \tag{5.2.4c}$$

When the proposed predictor is stable, all the LPM coefficients are under unity. As such the latter equation has a limit. In a practical sense this limit is reached very quickly and couples no valid meaning to the future confidence interval since the uncertainty or covariance of these predictor coefficients are not computed.

Stellenbosch University http://scholar.sun.ac.za

5.3 Safe Landing Opportunities

Typically the heave, pitch and roll states of the flight deck would be monitored and where their individual predicted safe-landing windows coincides would result in the master safe-landing window. For simplicity however, only the heave motion of the flight deck will used in the prediction algorithm to illustrate the concept. The criteria for a suitable landing window is before the peak in an ascending wave as shown in Figure 5.5. These regions are slow, rising and their acceleration faces downwards, translating into a minimal impact force on the landing gear of the helicopter. The ground effect on an upcoming flight deck will cushion the landing of the descending helicopter. These areas can be easily filtered out by taking the Euler backward approximation of the first and second order derivatives of the deck down position in the forecast window as seen in Equations 5.3.1a to 5.3.1b. It is important to note that $\dot{H_{max}}$ is the maximum allowable heave rate to tolerate for a safe landing window and ΔT is the predictor sample time. The numerous safe landing opportunities identified in the prediction horizon must also be larger than a predefined minimum landing window size to accommodate flight control transients whilst descending.

$$\frac{dH}{dt}(k) \approx \frac{H(k) - H(k-1)}{\Delta T} < \dot{H_{max}} \tag{5.3.1a}$$

$$\frac{d^2H}{dt^2}(k) \approx \frac{H(k) - 2H(k-1) + H(k-2)}{(\Delta T)^2} < 0 \tag{5.3.1b}$$

In noisy applications, the latter Euler backward approximations for the down velocity and acceleration respectively, will be inadequate. The LRLS LPM acts as a FIR filter and as such these approximations are only valid over a small period of time. Another very valuable trait of this LPM is its ability to provide a much better estimate for the current heave velocity of the flight deck, should the RUAV match the motion of the flight deck. The Euler central derivative approximation on 2 points is given by Eq. 5.3.2

$$\frac{dH}{dt}(k) \approx \frac{H(k+1) - H(k-1)}{2\Delta T} \tag{5.3.2}$$

This simple approximation can be made much more robust against sampling noise for a larger window size. The Euler central differentiation approximation for a window size of 8 points is given by Eq. 5.3.3

Stellenbosch University http://scholar.sun.ac.za

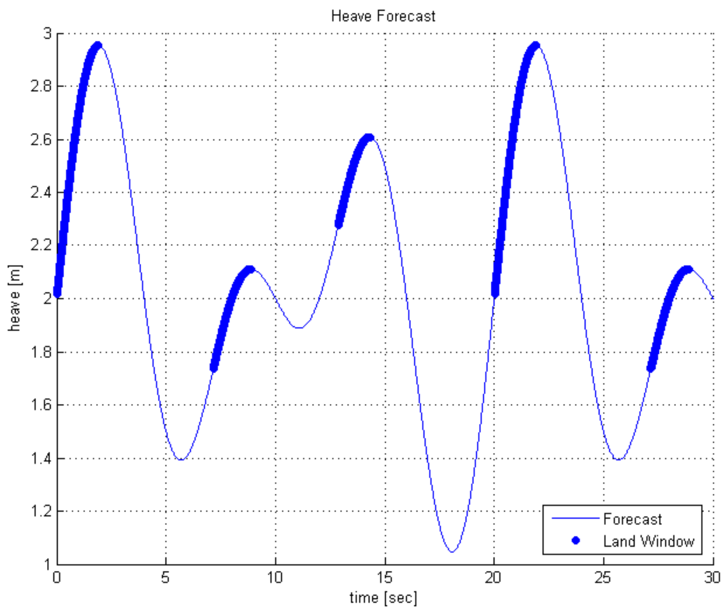


Figure 5.5 – Heave Forecast

$$\begin{aligned}\frac{dH}{dt}(k) \approx\ & \frac{14[H(k+1)-H(k-1)]+14[H(k+2)-H(k-2)]}{128\Delta T} \\ &+ \frac{6[H(k+3)-H(k-3)]+1[H(k+4)-H(k-4)]}{128\Delta T}\end{aligned} \tag{5.3.3}$$

5.4 Closure

The LLMS estimation scheme is utilized as the adaptation method to train the LPM parameters. The stability criterion for this method was analyzed to optimize the algorithm. The confidence in the LPM parameters was not modelled and inhibits this method to propagate on a confidence interval throughout the prediction horizon. Lastly, the criteria for safe landing opportunities were defined. These landing times must be incorporated into the landing controller for a controlled descent, as discussed in Chapter 6.

Stellenbosch University http://scholar.sun.ac.za

Chapter 6

Control System

The suitable landing opportunities identified in the prediction horizon, Chapter 5, is used in the landing controller to plan a controlled descent. A state-machine that enables smooth transitions between modes of motion is handled by the higher-level navigation outer-loop controllers designed by [9] and interfaces directly with the control loops. This project will use the existing architecture and append a new proposed landing controller that will be discussed in §6.2.

Apart from the precise timing of landing on a moving flight deck, one of the more challenging obstacles to overcome in a successful landing of any rotorcraft is ground effect. This phenomena and the implications it may hold in store for the heave controller, discussed §6.1, is discussed in §6.4. The horizontal position regulation of RUAV handled by the joint efforts of the longitudinal, lateral and heading controllers are assumed to be adequate. Subsequently, only the rotorcraft's heave controller will be discussed in §6.1. Finally, an overview is given on the implications that arise in the RUAV kinematic estimator of matching or following the heave motion of the flight deck in §6.3.

6.1 Heave Controller

The decoupled longitudinal, lateral, vertical and heading controllers that stabilize the RUAV in flight was designed by [5]. The latter PI controllers were further refined by [9] by analysing the frequency-domain response from actual flight-test data as described in [43]. This ensures better system identification that in turn leads to the design of tighter control loops for faster responses, without compromising system stability. The control system block diagram for the

Stellenbosch University http://scholar.sun.ac.za

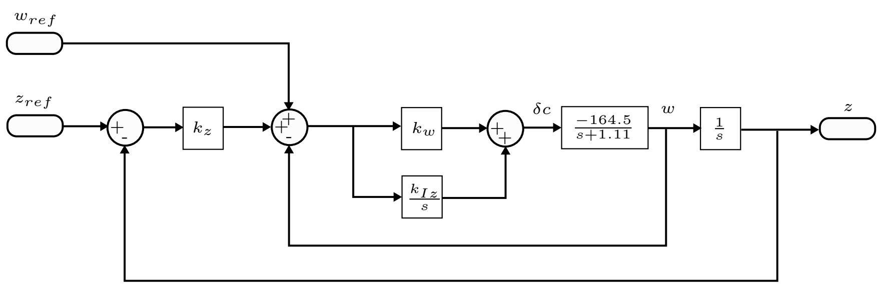


Figure 6.1 – Succesive loop closure of the heave-plant

successive loop closure heave controller is given in Figure 6.1. The heave plant is modelled by the transfer function, Equation 6.1.1.

$$\frac{w}{\delta c} = \frac{-164.5}{s + 1.11} \tag{6.1.1}$$

δc is the collective pitch angle in radians, w is the heave rate of the RUAV in m/s and z the heave position of the RUAV in m. $k_w = -0.02$, $k_{Iw} = -0.008$, $k_z = 0.5$ are the PI gains respectively. The closed loop transfer function, assuming no heave velocity feed forward ($w_{ref}(s) = 0$), gives rise to Equation 6.1.2. If the heave velocity of the flight deck is fed forward into the heave velocity controller ($w_{ref}(s) = sz_{ref}(s)$), the closed loop transfer function becomes Equation 6.1.3. The bode plot in Figure 6.2 shows a substantial increase in the bandwidth of the heave plant if velocity feed forward is utilized and forms a critical part of matching the heave of the flight deck. The bandwidth of $G1$ and $G2$ is 0.5349 rad/s and 2.9714 rad/s respectively.

$$G1(s) = \frac{z(s)}{z_{ref}(s)} = \frac{1.645s + 0.658}{s^3 + 4.4s^2 + 2.961s + 0.658} \tag{6.1.2}$$

$$G2(s) = \frac{z(s)}{z_{ref}(s)} = \frac{3.29s^2 + 2.961s + 0.658}{s^3 + 4.4s^2 + 2.961s + 0.658} \tag{6.1.3}$$

6.2 Descend Strategies

Prior to augmenting the existing navigation state machine with a low-level state to handle the landing algorithm, a comparison between the matching and linear landing approaches, as shown in Figure 6.3, with the emphasis on a vision system, is needed. In a linear descent, the RUAV will aim for a touchdown point in the prediction horizon and follow a linear

Stellenbosch University http://scholar.sun.ac.za

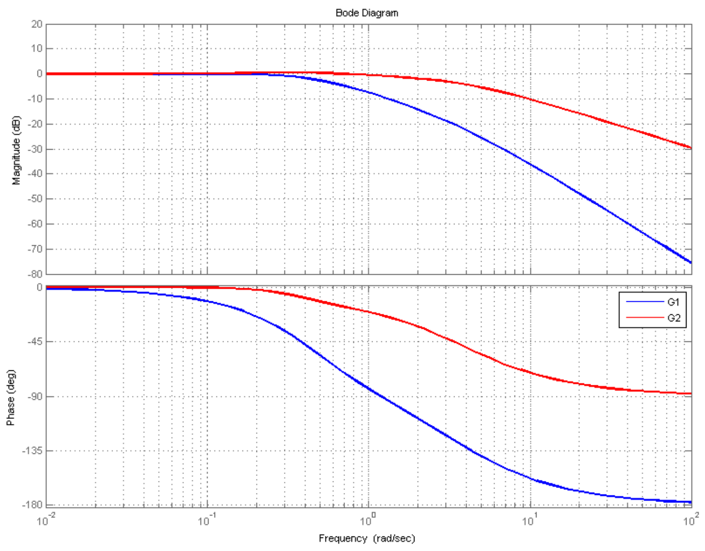


Figure 6.2 – Velocity Feed Forward Increased System Bandwidth

trajectory out of hover onto the flight deck. In a matching descent, the RUAV will aim for a touchdown point in a prediction horizon and correlate its own matching offset with respect to the flight deck closely to the time of touchdown. The following points of interest were taken into account.

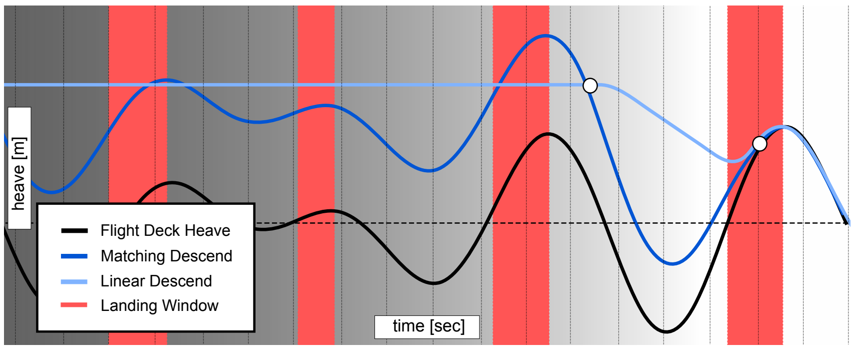


Figure 6.3 – Possible Descend Stategies

<u>**Visual Lock**</u>

The entry flight for a linear descent will maintain visual lock much longer than in a matching descent.

<u>**Ground Effect**</u>

As with the visual lock, the ground effect will be postponed until the very end of the linear descent. In matching mode however, ground effect will be engaged much earlier.

Stellenbosch University http://scholar.sun.ac.za

Control Effort

A linear descent is much easier to execute than a matching descent and will not provoke the cross-coupling modes of helicopter. This cross-coupling process noise will disturb the heave plant if the RUAV is stabilized by a slow control system.

Prediction Horizon

A linear descent relies heavily on both the phase and magnitude components of the prediction horizon. Phase is the only critical component for a matching descent since this mode will regulate its own hover height offset from the flight deck to the LT.

Collision Path

A linear descent needs to monitor possible intersection points from the predefined glide slope and the heave motion of the deck. A matching descent inherently avoids this phenomenon as it constantly maintains a hover offset above the deck.

The industry norm for a full scale helicopter landing on a heaving flight deck is that the pilot should match the heave motion of the flight deck. However, due to the severe limitations discussed in §6.3, a linear descent will be followed. Since the RUAV must first hover above the flight deck to populate a learning matching window, the new low-level state will be called Perching and the logical transitions within this state are shown in Figure 6.4.

The landing controller will use the prediction horizon to plan a linear glide slope with a defined descend gradient from the safe hover height to intersect the flight deck at the exact time of the LT. The RUAV's ability to monitor a future horizon, will renown this type of landing as an Oracle landing. The planned position of the RUAV on the glide slope will be fed into the heave position controller. The descend rate determines when the RUAV should climb onto the glide slope and only then will it be fed directly into the inner loop heave rate controller. To account for possible premature touchdowns, the instance of climbing onto the glide slope can be deliberately delayed or a minimum cushion height can be maintained as a clearance height above the flight deck. The latter cushion height will be used as a safeguard buffer against any possible unplanned collisions between the RUAV and the flight deck. The latter cushion height in conjunction with the optimal glide slope descend rate determines the internal Perching transition to a Forced landing mode. In this mode, pure heave velocity regulation is performed. All the loops of the longitudinal and lateral PI controllers are enabled till the touchdown of the RUAV to counter possible drift from the flight deck. The reference angles of the latter controllers are limited to be very small and as such would not pose as a serious problem to enable them until the touchdown of the RUAV. The transition

Stellenbosch University http://scholar.sun.ac.za

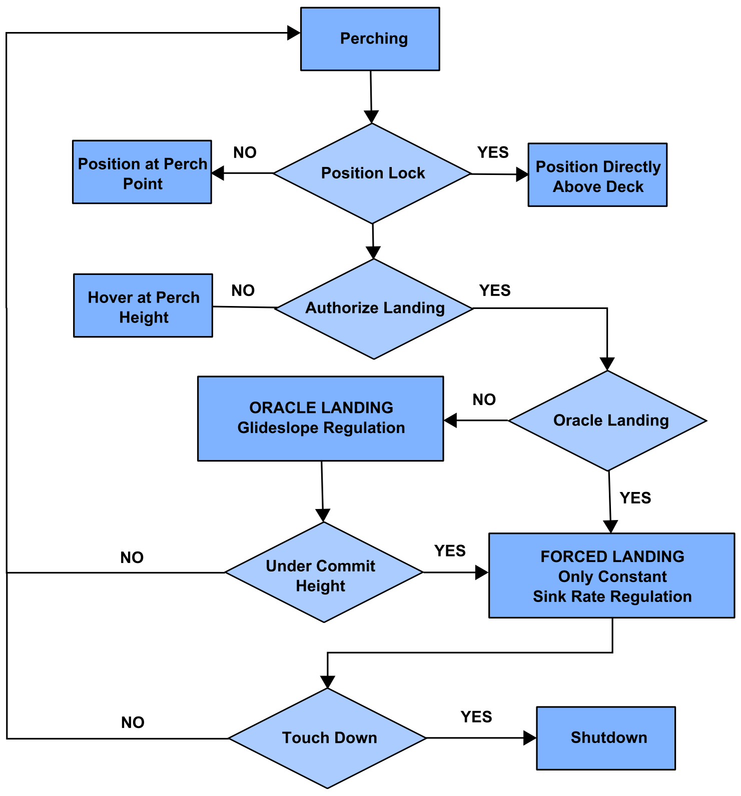


Figure 6.4 – Perching Landing Controller Architecture

from the Perching state to the Touchdown state is triggered by an accelerometer spike close to the vicinity of the flight deck. This low level state is responsible to ramp down and flush any residual control to their trim points and map the engine throttle to the idle trim.

6.3 Kinematic Estimator

The current onboard EKF estimator did not take into account the DGPS receiver latency, [44] [45], and is thus not in perfect synchronization with the faster IMU sensor telemetry. The effect of this latency is considered negligible in slow autonomous flight such as way-point navigation. In more aggressive flight like heave matching or following of the flight deck, the Down estimate, for example, is corrupted and lagged as the GPS Meters above sea level (MSL) and the double integrated acceleration measurement do not correlate to the same sample instant. A brief overview is given for the discrete state space position and velocity kinematic state estimator. In the absence of a measurement, $\mathbf{y}$, (GPS Position and Velocity), the previous best state can be propagated forward using the plant dynamics and the deterministic input, $\mathbf{u}$, (Accelerometers). This prediction step is shown in Equation

Stellenbosch University http://scholar.sun.ac.za

6.3.1. The error covariance of the propagated state grows as seen in Equation 6.3.2. The matrices, Q and R, represent the position/velocity process noise covariance and acceleration observation noise covariance.

$$\hat{x}^{-}_{pv}(k+1) = \Phi\hat{x}^{+}_{pv}(k) + \Gamma u(k) \tag{6.3.1}$$

$$P^{-}_{pv}(k+1) = \Phi P^{+}_{pv}(k)\Phi^{T} + Q(k) \tag{6.3.2}$$

When a measurement is available, the Kalman Filter gains can be calculated by Equation 6.3.3

$$L(k) = \frac{P^{-}_{pv}(k)H^{T}}{HP^{-}_{pv}(k)H^{T} + R(k)} \tag{6.3.3}$$

The innovation step is then applied to update the current best estimate is given by Equation 6.3.4. The covariance after the innovation step is given by Eq. 6.3.5.

$$\hat{x}^{+}_{pv}(k+1) = \hat{x}^{-}_{pv}(k) + L(k)[y_{pv}(k) - H\hat{x}^{-}_{pv}(k)] \tag{6.3.4}$$

$$P^{+}_{pv}(k+1) = [I - L(k)H]P^{-}_{pv}(k) \tag{6.3.5}$$

If the RUAV estimator states are combined with the camera measurements while matching the deck, the lag of the kinematic estimator will induce phantom dynamics of the flight deck and doing so corrupt the prediction algorithm. Figure 6.5 depict the estimate lag as it is aggravated with the increase of the match following frequency. To increase the bandwidth of the heave plant, the heave velocity of the flight deck can be fed forward directly into the heave rate controller, but these benefits can only be enjoyed once the estimator is fixed.

It is recommended to hold a regression history of past deterministic inputs and perform the estimator innovation step with the late state estimates. Forward propagation can then be applied on this history of inputs to provide an estimate of the current states and propagate the covariance. The latter update scheduled for the kinematic state estimator is not carried through due to the following reasons. The additional memory that needs to be allocated to buffer the history values of sensors can potentially provoke a stack buffer overflow and so a system reset in mid-flight. The numerous forward propagation steps can also pose as a

Stellenbosch University http://scholar.sun.ac.za

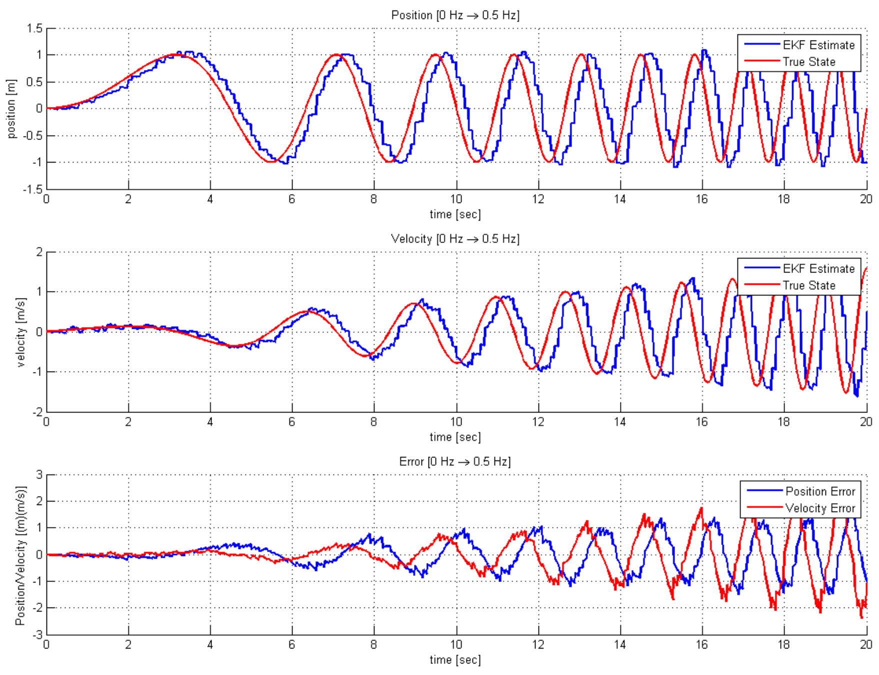


Figure 6.5 – Delay in Kinematic Estimator

timeout problem. Alternatively, the EKF can be augmented with a delay modelling (Pade approximation) of the DGPS measurements if the latency is known of the sensor. The EKF estimator delay is not deemed a problem for a slower linear descent of the helicopter.

6.4 Ground Effect

Ground effect is pronounced within one rotor length from any rotorcraft to the landing surface and results from an air cushion as the rotorcraft requires less collective pitch to remain in hover within closer proximity to the landing surface. Ground effect can be described by two distinct phenomena, [46]. Firstly, the forming of rotor-tip vortices is obstructed in close proximity to the ground. As such, the reduced induce airflow results in less drag that translates to faster blade rotations and more generated lift as seen in Figure 6.6. The second is that the flow of the downwash wind from the main rotor is obstructed close the ground surface. This causes an additional differential pressure that relates to larger lift forces.

The provisions made to attenuate the repercussions of ground effect are to descend with a faster sink rate. A more aggressive PI heave rate controller would integrate faster and

Stellenbosch University http://scholar.sun.ac.za

push quicker through ground effect. The pneumatic flight deck simulator skeleton is also expanded with a mesh grid landing platform. This avoids ground effect, as the downwash of the RUAV can pass through the flight deck.

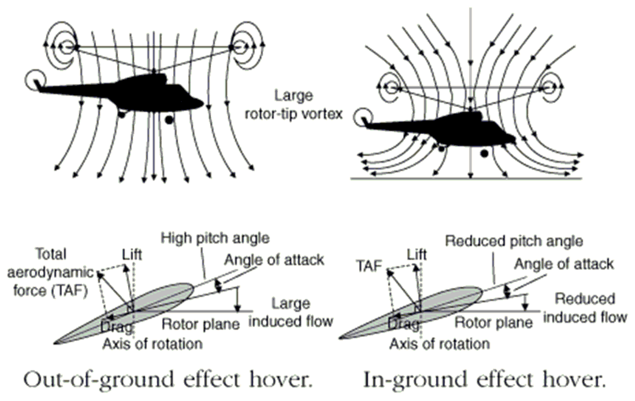


Figure 6.6 – Ground Effect [2]

6.5 Closure

The heave controller was investigated to analyze its performance in heave matching of the flight deck using heave velocity feed forward. Although, the industry norm for helicopter pilots are to match the heave of the flight deck prior to a touchdown, the delay in the EKF estimator proved a linear descent landing more viable. The design of new hardware to facilitate this landing controller scheme as well as the firmware that was implemented is discussed in Chapter 7.

Stellenbosch University http://scholar.sun.ac.za

Chapter 7

System Development

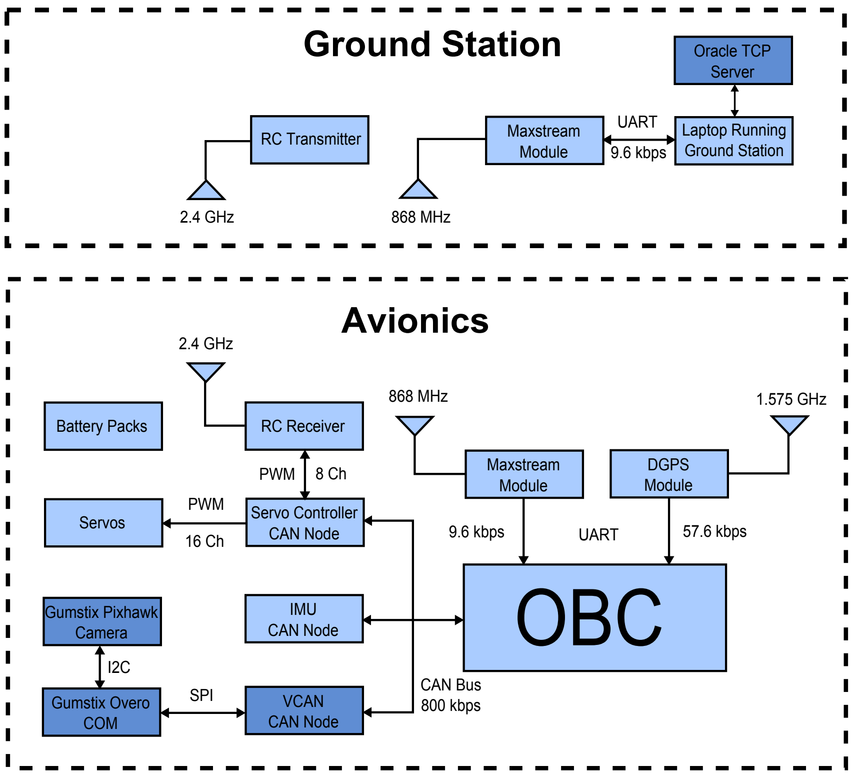


Figure 7.1 – System Architecture

This section elaborates on the hardware and firmware that was developed to implement the vision sensor. The backbone of the implementation of the vision algorithms rests on the processing power of a Gumstix Overo COM. Background on Gumstix and the attributes of the applicable module used for this thesis is discussed in §7.2. The new additions to the

Stellenbosch University http://scholar.sun.ac.za

existing avionics architecture is illustrated in Figure 7.1 in dark blue. Notice the Vision on Controller Area Network (VCAN) board that acts as the digital link between the OBC and the Gumstix COM. The development of the VCAN hardware and firmware is given in §7.1. Since vibration of any RUAV is renowned to pose as a serious image blur problem, vibration is roughly investigated in §7.3. Finally, a detailed overview is given on how the ground station interfaces with the Gumstix COM during flight.

7.1 VCAN Board

The Controller Area Network (CAN) bus is an ingenious protocol to share information across sensor nodes. The CAN bus is a popular avionics standard that allows multiple devices to exchange information on the same bus. Each information package is sent with a 16-bit extended identifier (EID). When a device notices a CAN system interrupt, applicable messages pertaining to 2 masks and a maximum of 6 filters, are loaded into the receive buffer, otherwise they are discarded. Packet conflict and data flooding of the CAN bus is resolved by assigning messages with a 4 rank priority.

The different Serial Peripheral Interfaces (SPI) operating voltages of the Gumstix COM (1.8V) and the OBC (4V) made direct, full-duplex communication impossible. Also, no definite communication standard interface with the Gumstix COM was defined for telemetry, as the SPI channels on both OBC microprocessors were occupied. The VCAN level-translator board was designed as a short-term solution to interface the Gumstix COM over SPI with the OBC via the CAN protocol. The VCAN board backbone stems from a dsPIC30F6014A microcontroller. The dsPIC32MX3XX/4XX Family datasheet Ref. [47] was used as a close guideline to configure the microprocessor appropriately. The Printed Circuit Board (PCB) layout schematics of the VCAN board is provided in Appendix B.

The timing diagram for the VCAN board is shown in Figure 7.2. The main program running on the Gumstix COM consists of the following sections. An almost instantaneous SPI duplex handshake between the VCAN board and the Gumstix COM will first exchange data, (1). Configure variables and RUAV estimator data will be decoded from the receive buffer to configure parameters and the camera driver, (2). The pixhawk section executing the computer vision follows immediately to provide an inertial estimate of the flight deck states, (3). The oracle section executing the motion prediction runs independently on a 10 Hz UNIX thread. It is vital that this section is synchronized to a fixed sample rate. The

Stellenbosch University http://scholar.sun.ac.za

main program will also not be able to reap the benefits of the OpenCV DSP acceleration should the oracle section be appended sequentially to the main program. After each program cycle, flight deck information is encoded into packages and passed to the transmit buffer, (4). The VCAN microprocessor slaves under the Gumstix COM SPI master. The VCAN clones its SPI receive buffer of flight deck estimates and then passes the labeled packages to the CAN bus, (5). In a similar fashion, a system interrupt downloads the RUAV state estimate CAN packages, (6) and assembles it for the SPI transmit buffer. By doing so, the VCAN board cloaks itself from the entire avionics since a direct connection between the Gumstix COM and the OBC can be established.

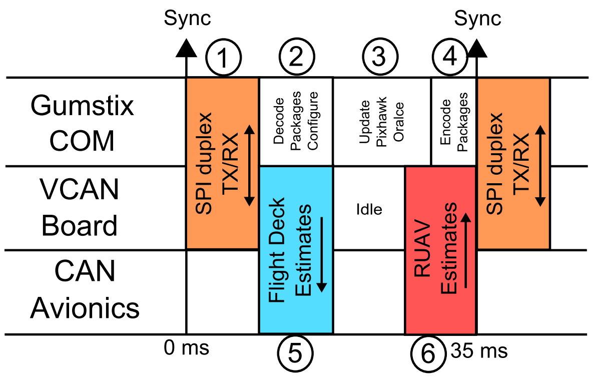
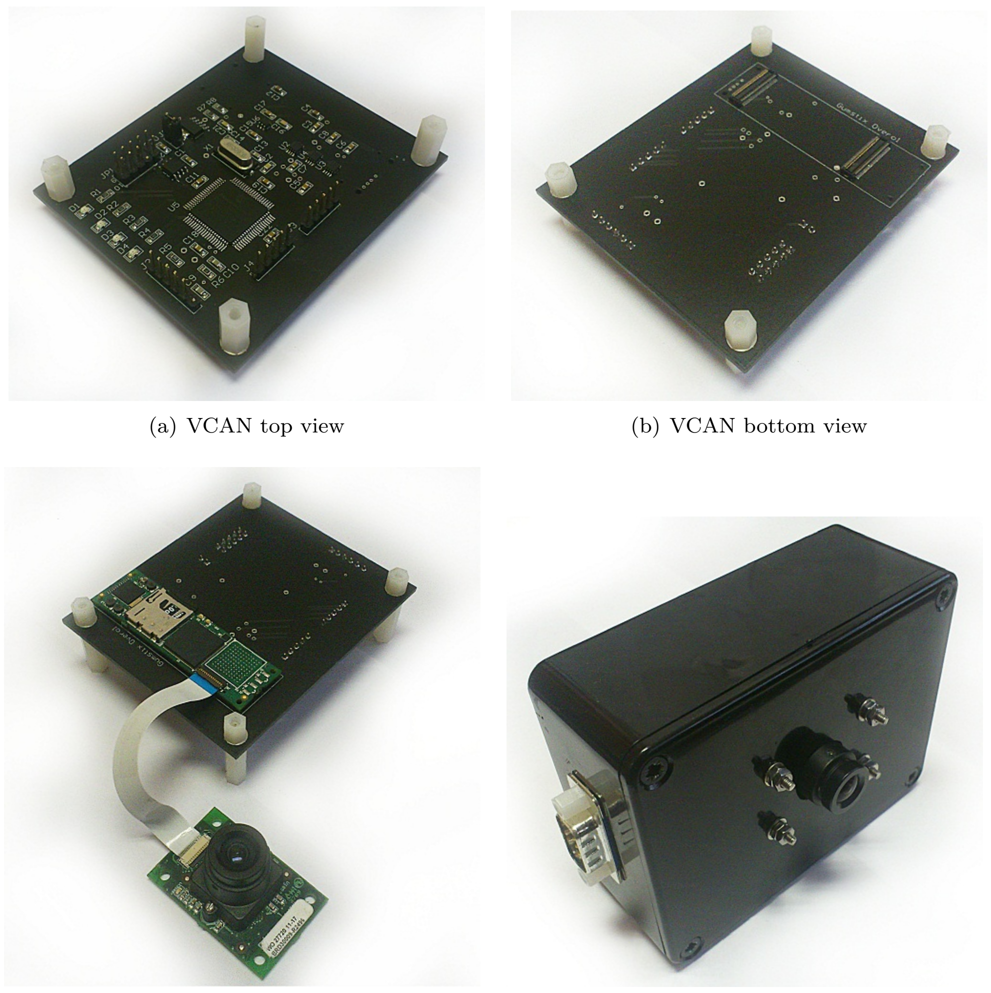


Figure 7.2 – VCAN Board Timing Diagram

Stellenbosch University http://scholar.sun.ac.za

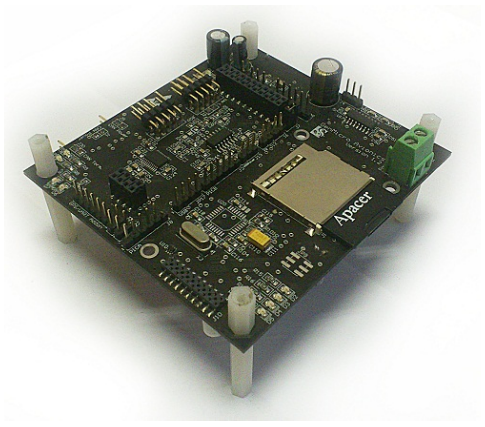

(a) VCAN top view (b) VCAN bottom view

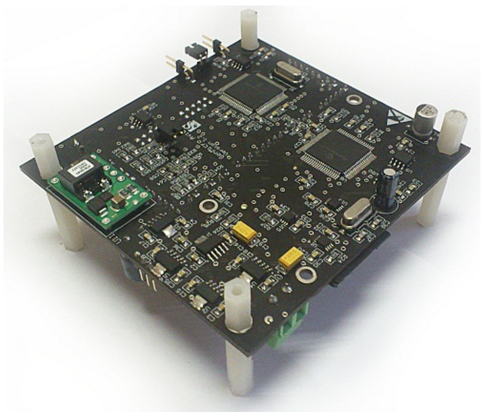
(c) VCAN bottom view (With Gumstix COM) (d) VCAN enclosed node

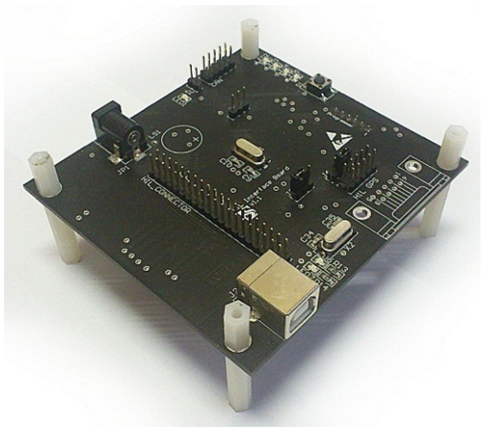
Figure 7.3 – VCAN Board

The VCAN board was programmed to maintain a 128-bit (8 x 16-bit words) transfer buffer to extract packages from the CAN bus with specific EIDs and them pass them consecutively the SPI handler. The latter board also maintained a 128-bit (8 x 16-bit words) receive buffer that would extract data from the SPI handler consecutively and load them onto the CAN bus with designated EIDs. The SPI master is set to a handshake data rate of 500 000 bits/s or 62.5 kB/s. This process is synchronized to the main program running at $\pm$ 30 Hz on the Gumtix COM. Every time the VCAN board receives the message identifier (0xAA55), it resets the package counter. See Appendix B for a detailed population of the TX/RX buffers.

Stellenbosch University http://scholar.sun.ac.za


(a) OBC TOP View


(b) OBC BOTTOM View


(c) HIL TOP View

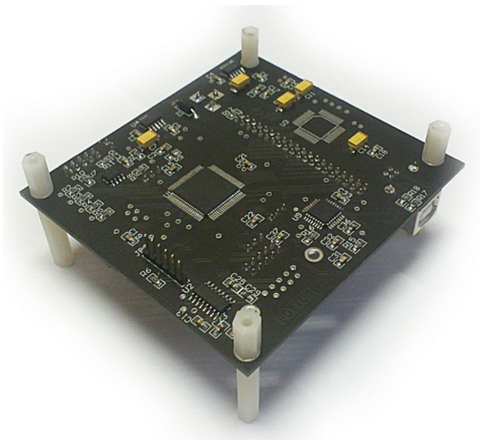

(d) HIL BOTTOM View

Figure 7.4 – OBC and HIL PCBs

7.2 Gumstix COM

7.2.1 Background

The Gumstix Company, which manufactures small, single-board computers, was founded by Gordon Kruberg in 2003, [48] [49]. The name Gumstix proved to be an appropriate name since the first computer produced was roughly the size of a stick of gum. The design of the primary computer boards have remained proprietary, expansion boards and their respective designs are published under a Creative Commons Share-alike license. The software stack is Linux based, built using the OpenEmbedded framework, [50].

At present, Gumstix has two product lines: the Texas Instruments OMAP-based Overo series and the Marvell XScale-based Verdex Pro series, [50]. A comprehensive range of

Stellenbosch University http://scholar.sun.ac.za

functions are made available through Gumstix products, including: OMAP, PXA, microSD, Bluetooth and 802.11g wireless interfaces, synchronous and asynchronous serial, USB, 10/100 Ethernet, RS-232 and more in a very small form factor. Linux for the OpenEmbedded build environment is made available by the company.

Numerous commercial, educational and hobbyist projects have used Gumstix products for developments such as power management metering devices, medical devices, security and personnel management products, wireless and hand-held products, unmanned aerial vehicles (UAV) and robotics. The OpenEmbedded software framework is implemented by Gumstix in order to track and fetch dependencies, cross-compile packages and build complete images by using BitBake, [51]. Once built, the rootfs image and the kernel are conveyed to the Gumstix through a serial connection, using compact flash or MMC type cards or through Ethernet network.

The OMAP3 processor onboard the Gumstix runs on a Linux kernel with Angstrom distribution as operating system. Embedded systems have become increasingly popular in the past couple of years. Subsequently the kernel, modules, boot loaders and root file systems can be compiled using BitBake, a make-alike build tool specifically focused on distributions and packages for embedded Linux cross compilation, [52]. Derived from Portage - the package management system used by the Gentoo Linux distribution - and then existing for some time in the OpenEmbedded project until it was separated in a standalone maintained, distribution-independent tool.

Writing custom Recipes which tell BitBake how to build a particular package ensures that Versatility is obtained. It includes all the package dependencies, sources from which to fetch the source code, configuration, compilation, build, install and remove instructions. The metadata for the package is also stored in standard variables. The BitBake Recipes are comprised of the source URL (http, https, ftp, cvs, svn, git, local file system location) of the package, dependencies and compile or install options. For the duration of the build process the BitBake Recipes are used to track dependencies, performing native or cross-compile of the package and pack it in a manner which is suitable to be installed on the local or target device. It is also possible to create complete images, consisting of a root file system and kernel. As a primary step in a cross build setup, the framework will attempt to create a cross-compiler tool chain acceptable for the target platform.

Stellenbosch University http://scholar.sun.ac.za

7.2.2 Kernel, Rootfs and Modules

In this thesis the native code compilation and OpenCV development packages were included. Furthermore, a custom module needed to be written to lend support to SPI device and the mt9v032 sensor of the Pixhawk camera. Unfortunately, the default release of the mt9v032 Linux drivers had numerous bugs within them. The author had to appropriately patch the source files of this driver and recompile the module. The corrections shown in Table 7.1 are considered of critical importance to complement the automatic filter. It is imperative to lock the physical gain and exposure levels of the camera else the bandwidth of the automatic filter would be very restricted and is likely to fail.

-	**Corrections**
1	Correction to the bad control indexing
2	Increased the maximum exposure level
3	Removal of automatic exposure and gain resets
4	Reordering of the query controls

Table 7.1 – MT9V032 patches

The OMAP3 processor will directly interface with the camera via an Inter-integrated Circuit (I2C) link. For development, the Gumstix COM is mounted on a Summit expansion board, which unfortunately lacked network capabilities. As such, communication between the Gumstix and a virtual Linux computer was done via a Kermit terminal interface. The device's boot loader was configured to automatically enter the main program after the boot script prior to the user login prompt. It would be impossible to pass this loop and for development, the following solution was proposed: The main program forces GPIO145 high (Pin 27 on 40-pin Header) and listens on GPIO146 (Pin 28 on 40-pin Header). The main program will only enter terminal mode if these two pins are short circuited with a jumper.

7.3 Vibration

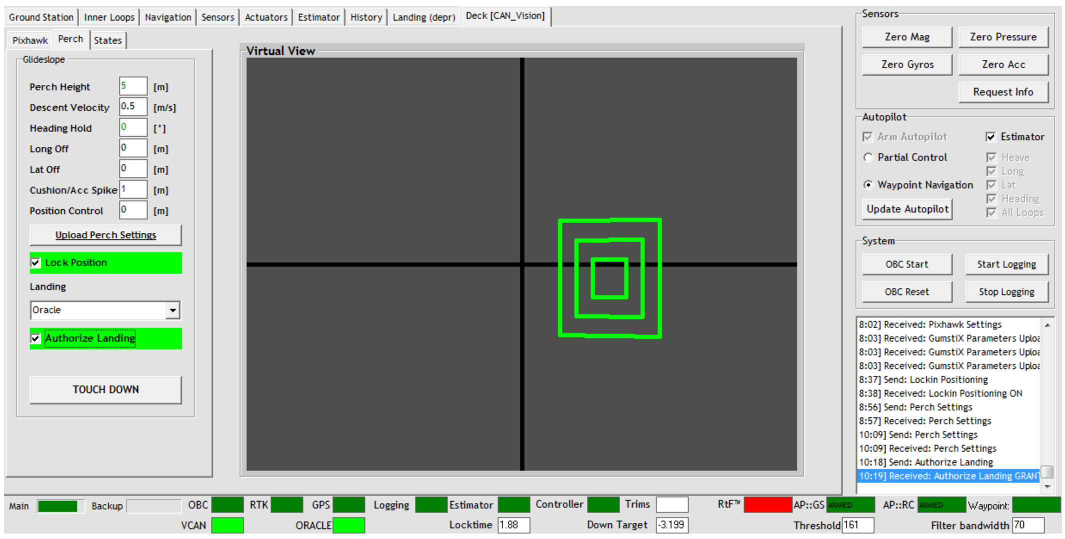
Vibration was roughly investigated to determine whether it would cause image-blur. Pre-recorded flight data was used to perform a FFT on the IMU telemetry to determine any dominant vibrations that the camera might experience, should it be mounted on the same anti-vibration mounts as the IMU. The existing setup showed no dominant peaks and considering the very fast global shutter (TrueSNAP) speed of 60 Hz of the camera, vibration would not pose to be a problem. The new silicone anti-vibrasion mounts that were moulded is shown in Figure 7.5. The aerial photos displayed in Chapter 9 also validates the success of these mounts.

Stellenbosch University http://scholar.sun.ac.za

Figure 7.5 – Anti-vibration Mount

7.4 Ground Station

Although this project is an autonomous system, it is still commanded and monitored by the Ground Station (GS). The GS provides a direct link to the RUAV to monitor vitals and upload commands. The existing GS was updated to accommodate the vision system as shown in Figure 7.6. The limited data link architecture crippled the option to stream down the live images from the Gumstix camera to the GS. The relative states are reconstructed and are used in a graphical re-projection of the image markers to give a bird's eyes view from the RUAV as display the view as seen from the Gumstix camera which is of great benefit to the GS operator in monitoring the landing process.



Glideslope

Perch Height	5	[m]
Descent Velocity	0.5	[m/s]
Heading Hold	0	[°]
Long Off	0	[m]
Lat Off	0	[m]
Cushion/Acc Spike	1	[m]
Position Control	0	[m]

✓ Lock Position

Landing: Oracle

✓ Authorize Landing

Autopilot

✓ Arm Autopilot
✓ Estimator
○ Partial Control
✓ Heave
✓ Long
● Waypoint Navigation
✓ Lat
✓ Heading
✓ All Loops

```
8:02] Received: Pixhawk Settings
8:03] Received: GumstiX Parameters Uploa
8:03] Received: GumstiX Parameters Uploa
8:03] Received: GumstiX Parameters Uploa
8:37] Send: Lockin Positioning
8:38] Received: Lockin Positioning ON
8:56] Send: Perch Settings
8:57] Received: Perch Settings
10:09] Send: Perch Settings
10:09] Received: Perch Settings
10:18] Send: Authorize Landing
10:19] Received: Authorize Landing GRANT
```

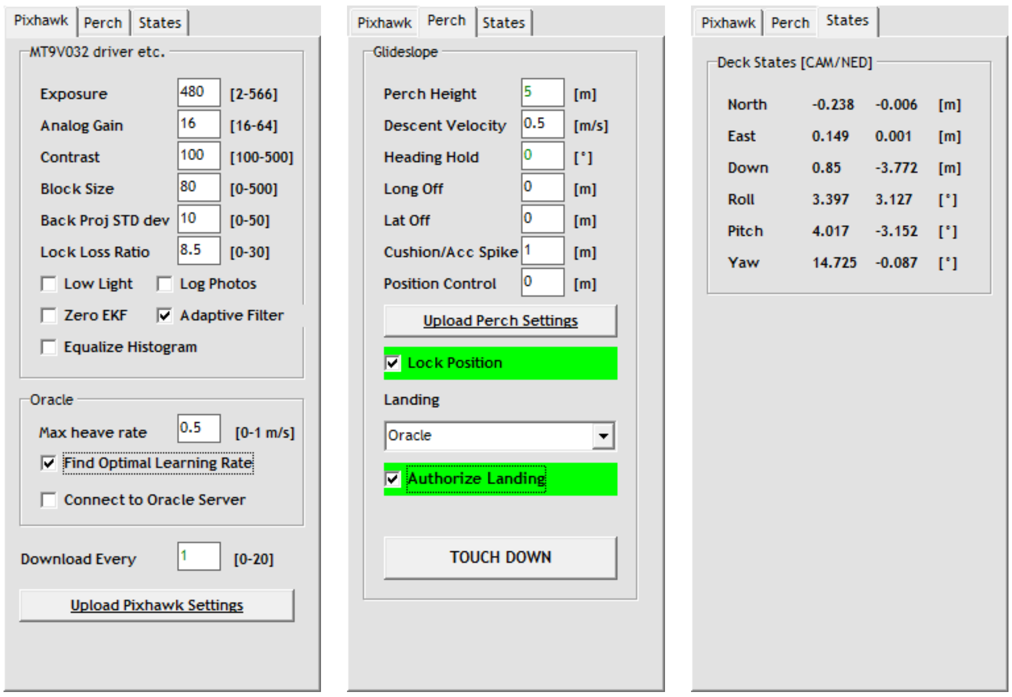
Main 🟩 Backup OBC 🟩 RTK 🟩 GPS 🟩 Logging 🟩 Estimator 🟩 Controller 🟩 Trims RtF™ 🟥 AP::GS ARMED AP::RC ARMED Waypoint: 🟩

VCAN 🟩 ORACLE 🟩 Locktime 1.88 Down Target -3.199 Threshold 161 Filter bandwidth 70

Figure 7.6 – Deck Vision Control

The reprojected concentric squares will become green if VL is obtained. If VL is lost, the last inertial position of the flight deck is held and the the relative states are re-calculated

Stellenbosch University http://scholar.sun.ac.za



MT9V032 driver etc.

Exposure	480	[2-566]
Analog Gain	16	[16-64]
Contrast	100	[100-500]
Block Size	80	[0-500]
Back Proj STD dev	10	[0-50]
Lock Loss Ratio	8.5	[0-30]

☐ Low Light ☐ Log Photos
☐ Zero EKF ☑ Adaptive Filter
☐ Equalize Histogram

Oracle

Max heave rate 0.5 [0-1 m/s]
☑ Find Optimal Learning Rate
☐ Connect to Oracle Server

Download Every 1 [0-20]

Upload Pixhawk Settings

Glideslope

Perch Height	5	[m]
Descent Velocity	0.5	[m/s]
Heading Hold	0	[°]
Long Off	0	[m]
Lat Off	0	[m]
Cushion/Acc Spike	1	[m]
Position Control	0	[m]

Upload Perch Settings

☑ Lock Position

Landing

Oracle

☑ Authorize Landing

TOUCH DOWN

Deck States [CAM/NED]

North	-0.238	-0.006	[m]
East	0.149	0.001	[m]
Down	0.85	-3.772	[m]
Roll	3.397	3.127	[°]
Pitch	4.017	-3.152	[°]
Yaw	14.725	-0.087	[°]

(a) Pixhawk Tab (b) Perching Tab (c) States Tab

Figure 7.7 – Deck Vision Tabs

using the latest RUAV inertial states. When this happens, the geometric pattern will become red. All the uploadable parameters for the vision system is displayed in Table 7.2. The uploadable parameters that regulated the landing controller is given in Table 7.3.The vitals applicable to the vision system which are monitored during flight, is displayed in Table 7.4.

Parameter	**Value**	**Comment**
Exposure	2-566	-
Analog Gain	16-64	-
Contrast	100-500	-
Blocksize	0-500 [mm]	Smallest square quadrant point dimension
Back Proj STD dev	0-50 [pixels]	-
Lock Loss Ratio	0-30	Hover height ratio
Low Light	0/1	-
Log Photos	0/1	1 Hz
Zero EKF	0/1	Return relative flight deck state information
Adaptive Filter	0/1	-
Equalize Histogram	0/1	-
Find Optimal Learning Rate	0/1	-
Download Package Rate	0-20	Minimize the load on the RF link

Table 7.2 – Uploadable Vision Parameters

Stellenbosch University http://scholar.sun.ac.za

Parameter	**Value**	**Comment**
Perching		
Perch Height	± 15 [m]	-
Descend Velocity	0-5 15 [m/s]	Glide slope gradient
Heading Hold	± 360 [deg]	Land heading
Long Off	± 15 [m]	Longitudinal step command
Lat Off	± 15 [m]	Lateral step command
Acc Spike Height	0-5 [m]	Distance from the flight deck to expect touchdown accelerometer spike
Critical		
Position Lock	0/1	-
Landing Type	NONE/Forced/Oracle	-
Authorize Landing	0/1	-

Table 7.3 – Uploadable Perching/Critical Parameters

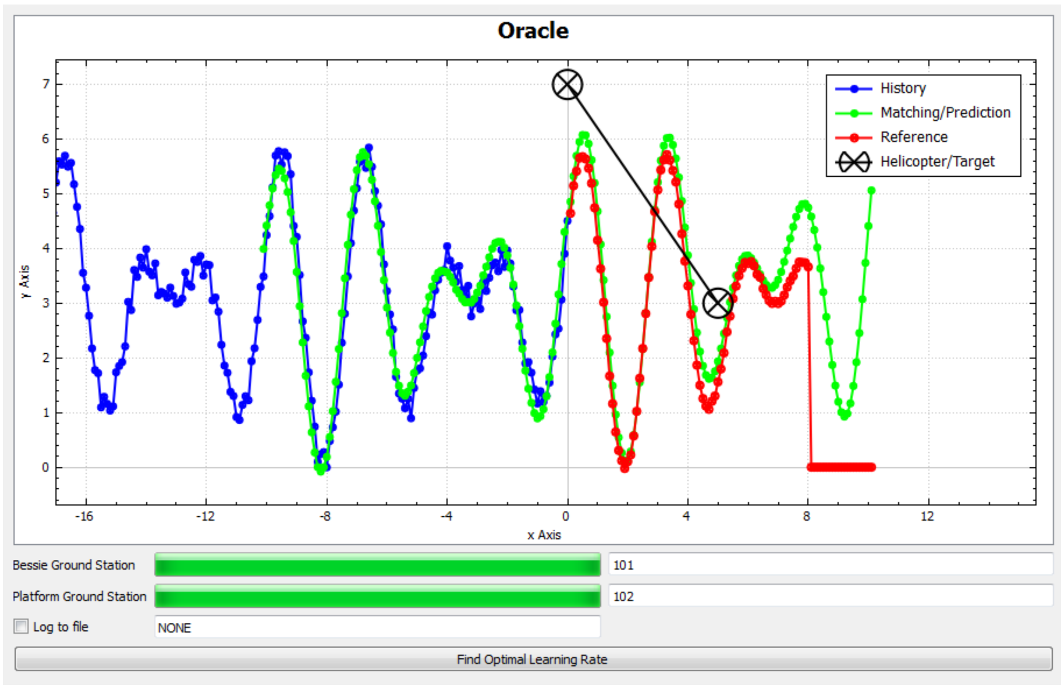
Parameter	**Value**
Vision	
Threshold	0-255
Filter Bandwidth	0-100
Squares Locked	0-3
Virtual View	-
Landing Controller	
Deck States [CAM/NED]	-
Lock in	0-20 [sec]
Deck Down Target	± 15 [m]

Table 7.4 – Monitors

Since timing is such a critical component in a safe, precision landing, the GS interfaces with a programmed Transmission Control Protocol (TCP) server, Figure 7.8. The main purpose of this TCP server is to emulate what the Gumstix COM would have seen that could not be streamed down due to transmission limitations. As such, this server executes a clone of motion prediction algorithm running on the Gumstix COM to emulate the prediction horizon cast be the Gumstix COM. The LT and target heave of the predicted point of impact is only streamed down from RUAV and could be overlayed on the TCP prediction horizon to double check whether that point indeed correlates to a safe landing.

Another critical component of this server is it draws the future reference heave data from the 3-DOF pneumatic platform GS TCP client into the monitor window. The redundancy to trigger on a false landing is vastly increased by doing so.

Stellenbosch University http://scholar.sun.ac.za



Bessie Ground Station 101

Platform Ground Station 102

☐ Log to file NONE

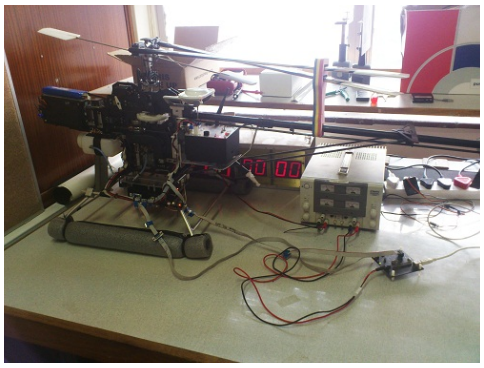
Figure 7.8 – Oracle Server

Stellenbosch University http://scholar.sun.ac.za

Chapter 8

Hardware in the Loop Simulations

HIL simulations are the remaining bridge to cross before actual flight tests. This test bed allows the non-linear *Simulink*^{TM} simulation to interface directly with the hardware onboard the RUAV on the day of flight, shown in Figure 8.1. Since the avionics will be stationary for HIL simulations the IMU (accelerometer, gyro, pressure sensor, magnetometer) and GPS virtual measurements are generated and passed to the OBC as shown in Figure 8.2. By doing so, the actual OBC outer-loop navigation and inner-loop controllers are tested.


(a) HIL Hardware Component

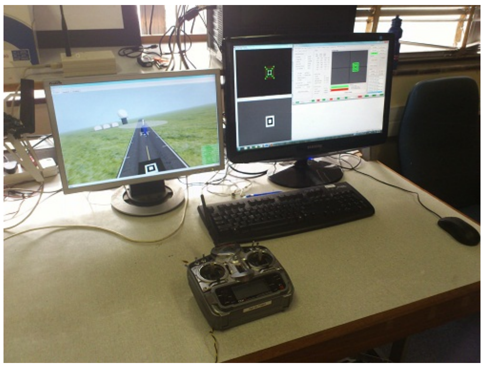

(b) HIL Software Component

Figure 8.1 – Hardware in the Loop Setup

For this particular project, the existing HIL firmware on the HIL board had to be updated to accommodate the CAN packets transmitted and received from the Gumstix COM. These extra packets are clearly marked in Figure 8.3. The native C source file for the MEX S-Function on the *Simulink*^{TM} side also had to be modified to accommodate this update. This Gumstix COM emulation block set takes the real states of the flight deck and the

Stellenbosch University http://scholar.sun.ac.za

helicopter to generate a virtual image as described by the intrinsic and distortion calibrated values as seen in Figure 8.5. OpenCV libraries are included to physically generate a photo.

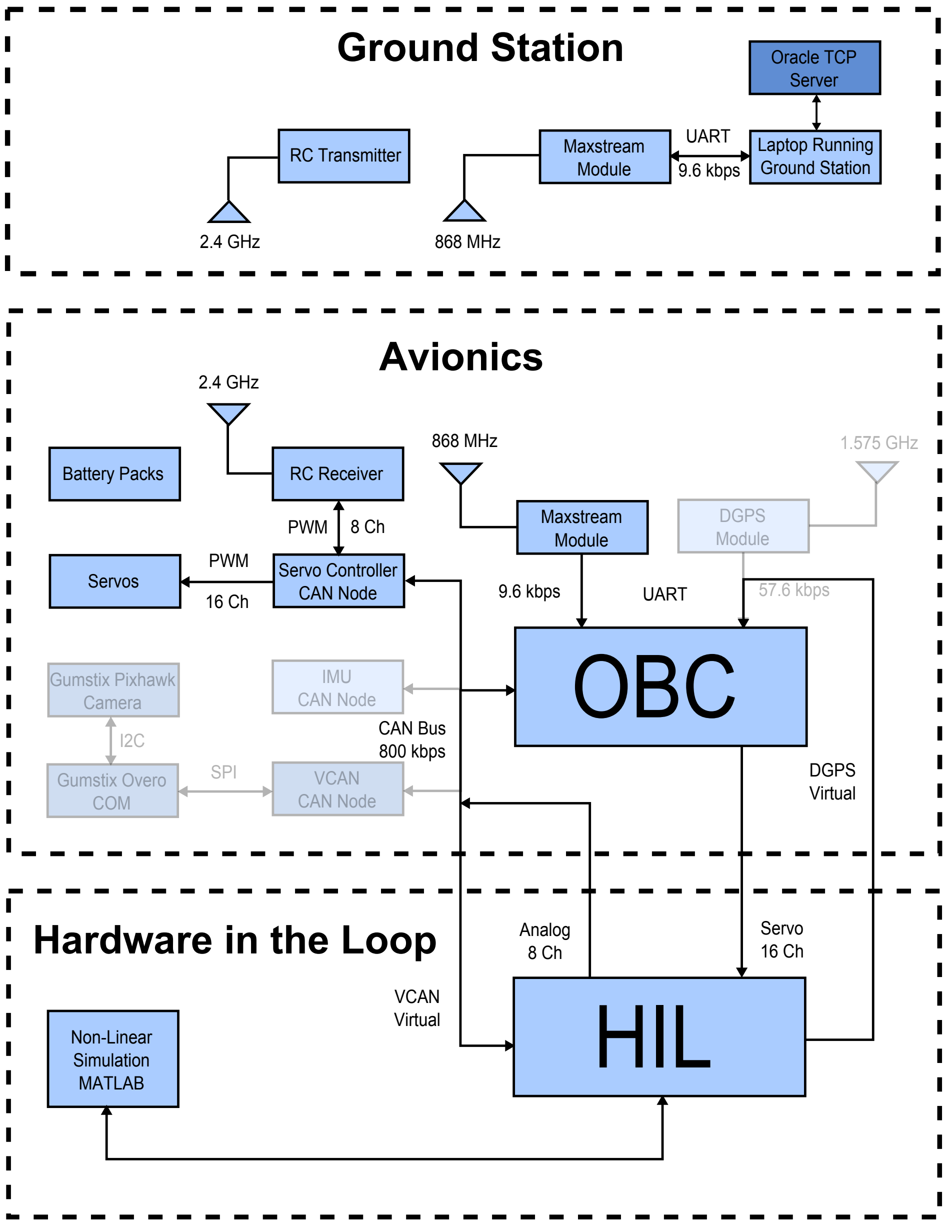


Figure 8.2 – Hardware in the Loop Architecture

The pixhawk section performs the pattern recognition and pose estimation on this photo. The flight deck states are then updated and also injected into the oracle section for the motion prediction. In HIL mode, the information generated from the Gumstix emulation

Stellenbosch University http://scholar.sun.ac.za

block set in $MATLAB^{TM}$, is passed to the HIL board, where it is emitted onto the CAN bus, to mimic the VCAN board.

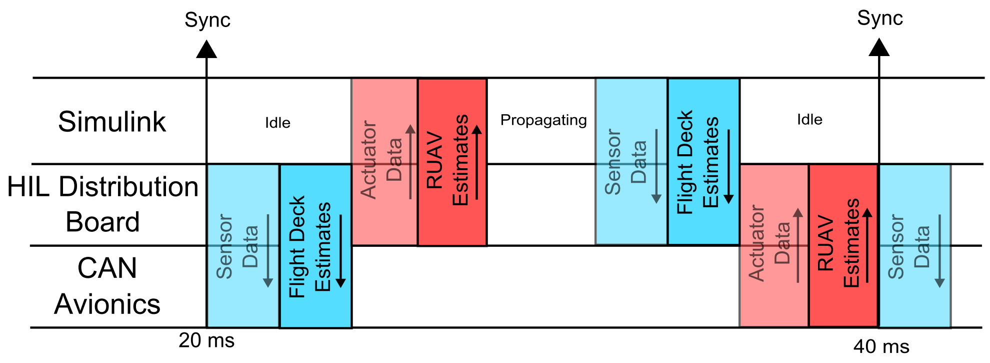
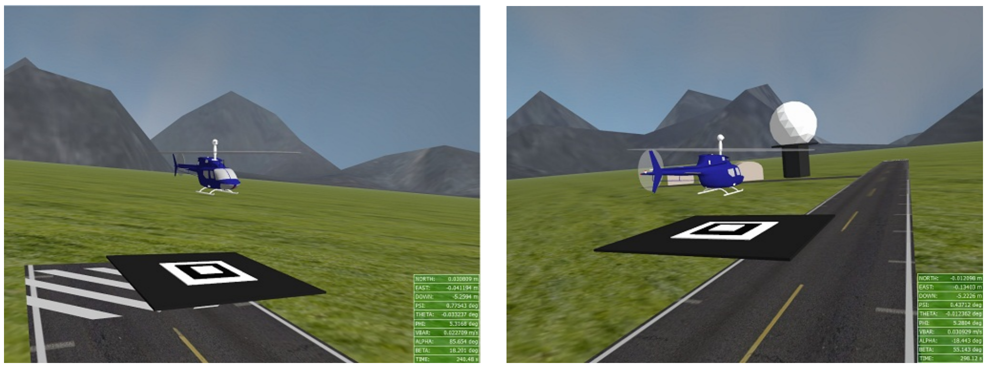


Figure 8.3 – HIL Board Timing Diagram

Great emphasis is placed on the collision dynamics of the flight deck on the helicopter skids upon touchdown. As such, a new collision model, derived in Appendix A, was developed to model the dynamic interaction between the helicopter skids and the flight deck. One of the biggest benefits of this HIL environment is that true states of the RUAV and the flight deck, from the non-linear $MATLAB^{TM}$ simulation, is streamed over TCP to a QT OpenGL visualization application, shown in Figure 8.4. The re-projection of the flight deck in a virtual virtual photo, the processed image markers and the motion prediction history, shown in Figure 8.5, is also displayed to complement the visualization of the simulation.

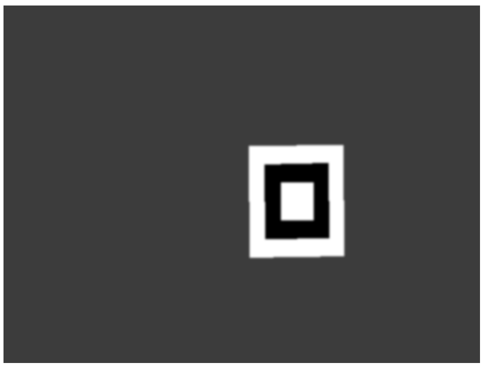

```
NORTH:     0.030809 m
EAST:     -0.041194 m
DOWN:       -5.2994 m
PSI:      0.77543 deg
THETA:  -0.033237 deg
PHI:       5.3168 deg
VBAR:    0.022709 m/s
ALPHA:     85.654 deg
BETA:      18.201 deg
TIME:        240.48 s
```

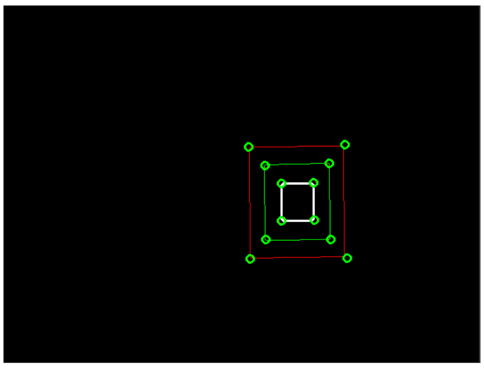
```
NORTH:    -0.012098 m
EAST:      -0.13403 m
DOWN:       -5.2226 m
PSI:      0.43712 deg
THETA:  -0.012362 deg
PHI:       5.2804 deg
VBAR:    0.030929 m/s
ALPHA:    -18.443 deg
BETA:      55.143 deg
TIME:        298.12 s
```

(a) (b)

Figure 8.4 – QTGL Engine Visualization Screenshots

Stellenbosch University http://scholar.sun.ac.za


(a) Virtual Photo


(b) Processed Image Markers

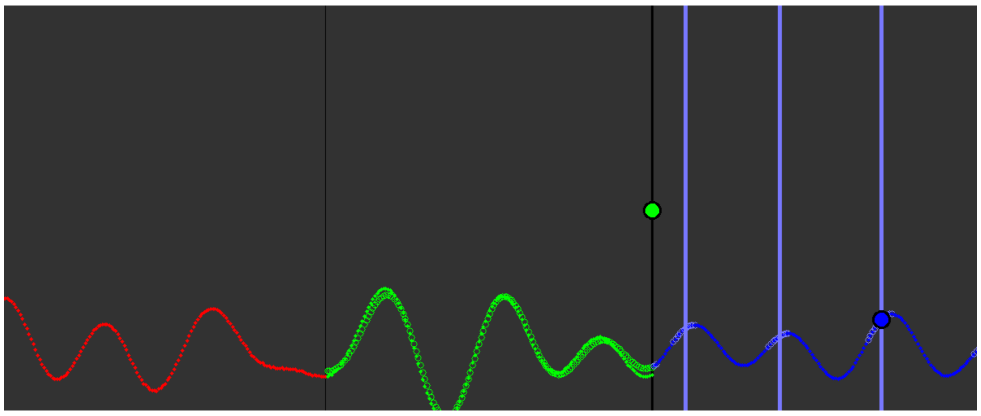

(c) Oracle History

Figure 8.5 – HIL Auxiliary Screens

Stellenbosch University http://scholar.sun.ac.za

8.1 Stationary Flight Deck

8.1.1 HIL Flight Test 1

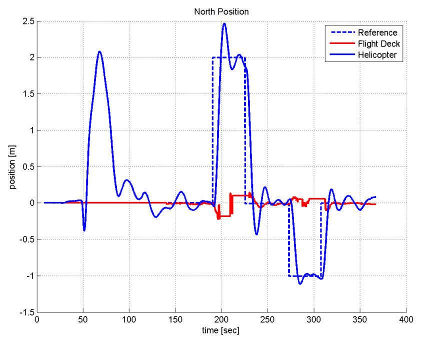
Timestamp	**Event**	**Value**
140 sec	Automatic Filter	Triggered
190 sec	Long,Lat,Heave Step Command	[2,2,-5]
211 sec	Long,Lat,Heave Step Command	[2,2,-4]
225 sec	Long,Lat,Heave Step Command	[0,0,-4]
273 sec	Long,Lat,Heave Step Command	[-1,-1,-3]
293 sec	Long,Lat,Heave Step Command	[-1,-1,-2]
308 sec	Long,Lat,Heave Step Command	[0,0,-2]

Table 8.1 – HIL Flight Test 1 Transitions

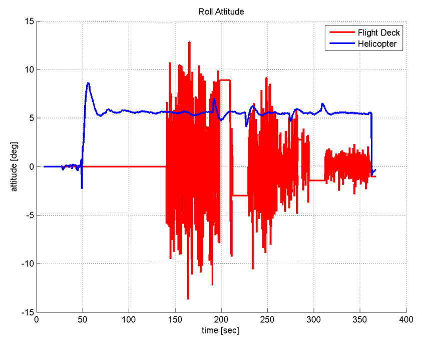
This simulation was conducted for a stationary flight deck and is in reference to the flight log in Figure 8.6. A number of heave, longitudinal and lateral step commands are issued to the RUAV to validate the accuracy of the VCAN as shown in Table 8.1.

Notice the significant increase in accuracy of the flight deck attitude measurements when the RUAV came closer to the flight deck. The flat regions in the estimates are where VL was lost since a ZOH principal is applied to the flight deck inertial estimates. The small bias on the down estimate of the flight deck is due to the 27 cm offset from the NovAtel Receiver to the landing gear of the RUAV. When the RUAV was initialized on the flight deck, the actual pattern was located 27 cm downwards. The sensor noise of the VCAN node is well represented by the noise lattice graphs discussed in §4.4.

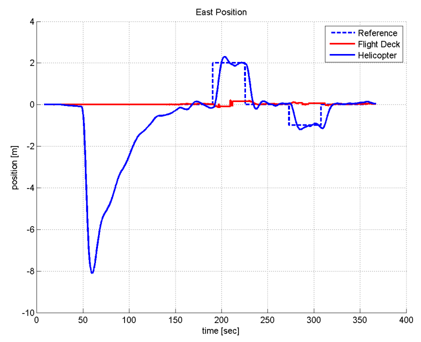
Stellenbosch University http://scholar.sun.ac.za



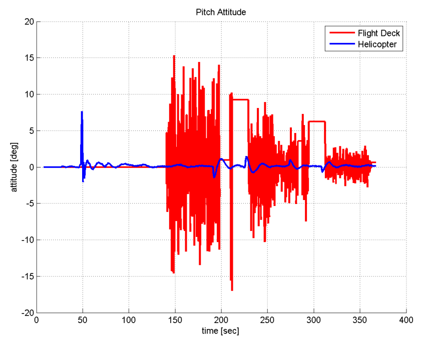
(a) North Position



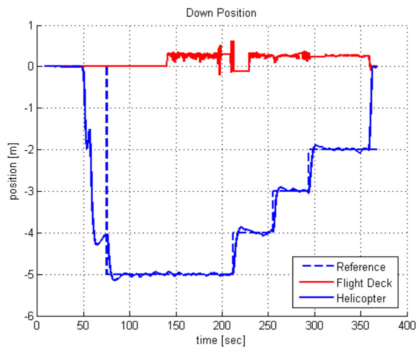
(b) Roll Attitude



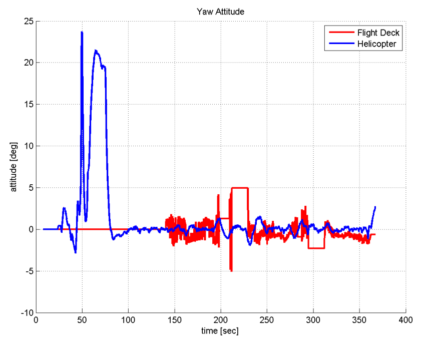
(c) East Position



(d) Pitch Attitude



(e) Down Position



(f) Yaw Attitude

Figure 8.6 – HIL Flighttest 1 : Flight Log

Stellenbosch University http://scholar.sun.ac.za

8.1.2 HIL Flight Test 2

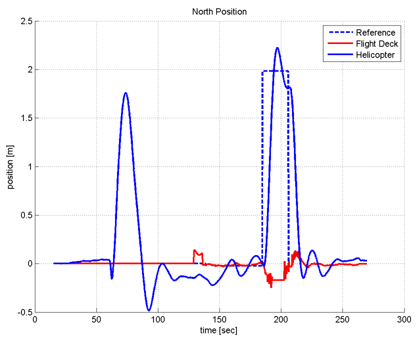
Timestamp	**Event**	**Value**
129 sec	Adaptive Filter	Triggered
190 sec	Long,Lat,Heave Step Command	[2,2,-5]
206 sec	Position Lock	Enabled
252 sec	Forced Landing	Authorized
264 sec	Touchdown	-

Table 8.2 – HIL Flight Test 2 Transitions

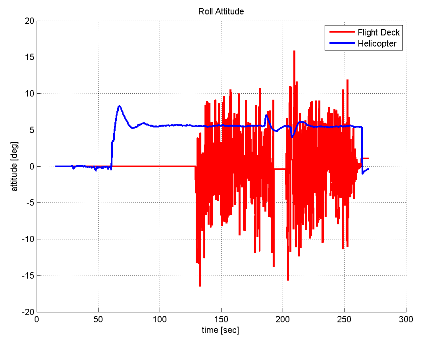
This simulation was conducted for a stationary flight deck and is in reference to the flight log in Figure 8.7. The position-lock of the Perching low-level state was tested in this test.

Notice the flight deck estimate variance especially large in the attitude measurements as the hover height ratio (refer to Chapter 4) is considered very large. This very conservative approach ensures that the geometric patten of the flight deck is located at the predefined hover height of 5m. The phantom flight deck dynamics caused by the latency in the RUAV EKF estimator, as described in §6.3, can be seen clearly in Figure 8.7, after position-lock was enabled and during the landing decend of the RUAV.

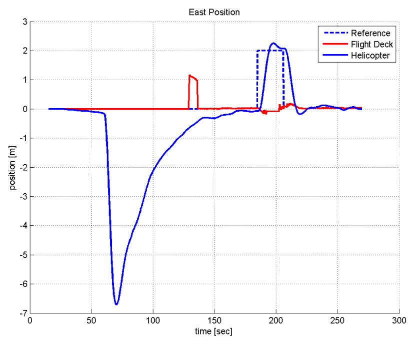
Stellenbosch University http://scholar.sun.ac.za



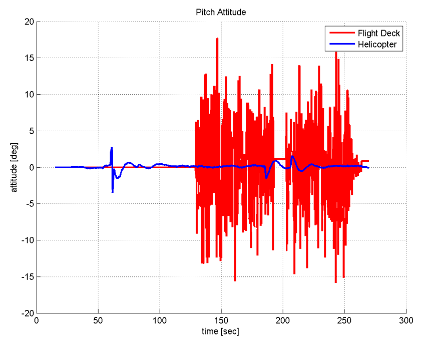
(a) North Position



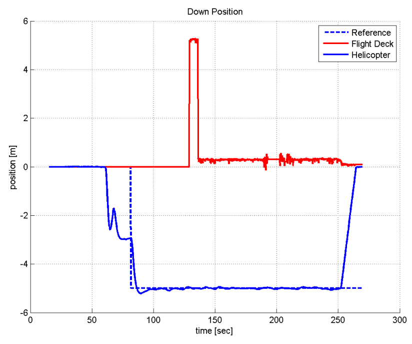
(b) Roll Attitude



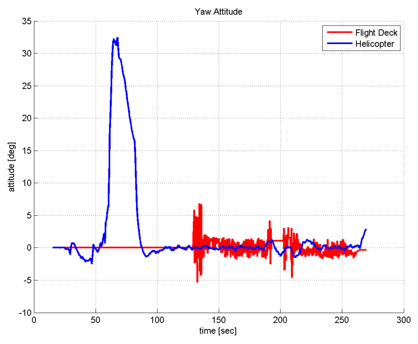
(c) East Position



(d) Pitch Attitude



(e) Down Position



(f) Yaw Attitude

Figure 8.7 – HIL Flighttest 2 : Flight Log

Stellenbosch University http://scholar.sun.ac.za

8.2 Heaving Flight Deck

8.2.1 HIL Flight Test 3

Timestamp	Event	Value
113.3 sec	Adaptive Filter	Triggered
115.8 sec	Position Lock	Enabled
165.1 sec	Find Optimal Learning Rate	Triggered
202.8 sec	Oracle Landing	Authorized
216.5 sec	Forced Landing	Commit
217.6 sec	Touchdown	-

Table 8.3 – HIL Flight Test 3 Transitions

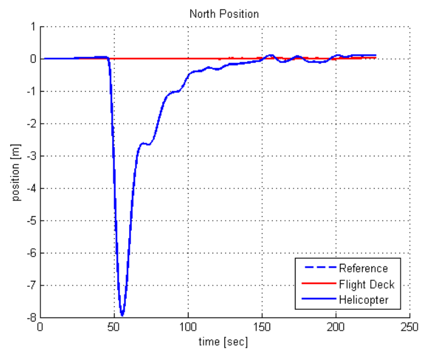
This simulation was conducted for a heaving flight deck and is in reference to the flight logs Figures 8.8 and 8.9. The Oracle landing mode of the Perching low-level state was tested in this simulation. The RUAV was brought to a still hover over the flight deck in a similar fashion as the previous simulations. Once VL was obtained on the flight deck, position-lock was activated to force the RUAV to track the horizontal position of the flight deck at a safe hover offset of 4 m above the flight deck. During this Perching phase, the learning window for the prediction algorithm is populated prior to the forecast of possible landing zones.

Notice from Figure 8.9 that the prediction algorithm will continuously stream down the furthermost suitable LT of the whole prediction horizon. The prediction algorithm only latches onto current LT as soon as the landing authorization command is given. This LT and the predicted heave of the flight deck that correlates to this point in time, are sent to the Oracle landing controller to plan a glide slope from the perching hover point to the forecast touchdown point. The position and velocity of this glide slope is fed directly into the heave and the heave rate controllers. A predefined optimal descend velocity, (0.5 m/s for this simulation) determines when the RUAV mounts the glide slope. A predefined commit window (1 sec for this simulation) determines when this landing controller transitions to a Forced landing, where pure heave velocity is regulated prior to touchdown with the flight deck.

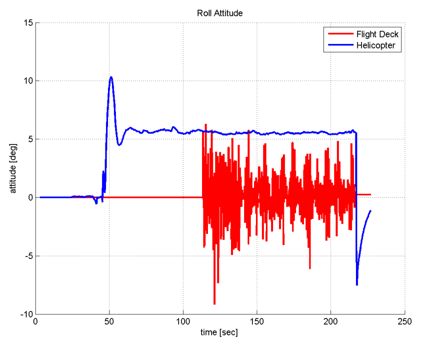
Authorized Landing Commmand [1]

The authorized landing command for an Oracle landing is issued at this instance. The prediction algorithm will latch unto the farest safe landing window within the prediction horizon and continuously monitor the predicted state information at this receding time.

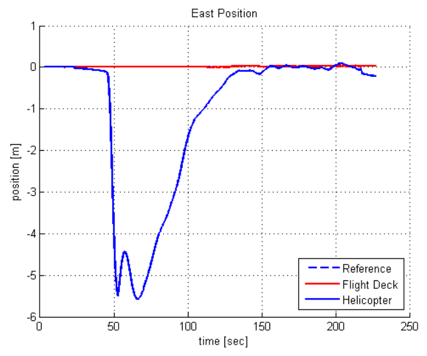
Stellenbosch University http://scholar.sun.ac.za



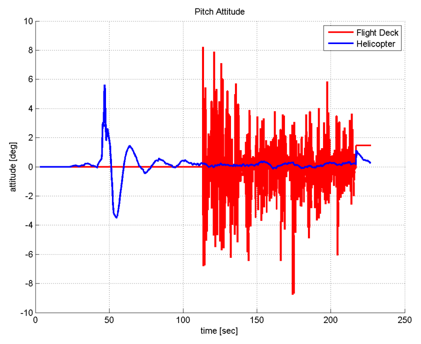
(a) North Position



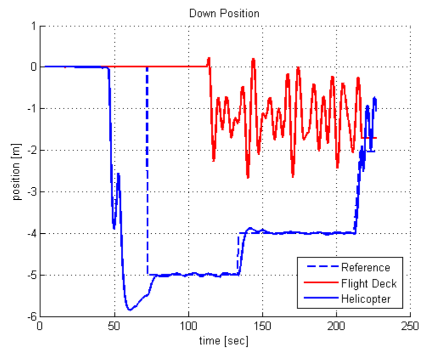
(b) Roll Attitude



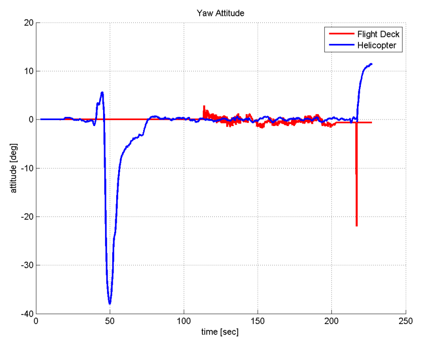
(c) East Position



(d) Pitch Attitude



(e) Down Position



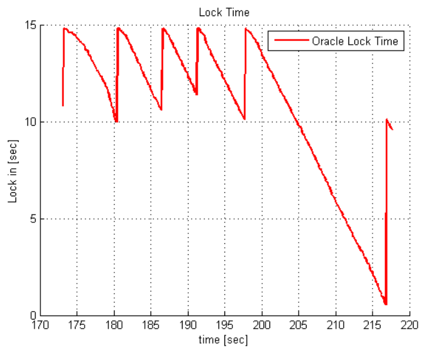
(f) Yaw Attitude

Figure 8.8 – HIL Flighttest 3 : Flight Log

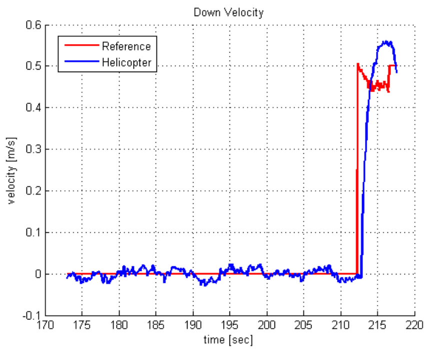
Mounting the Landing Glideslope [2]

The glide slope planned by the landing controller is dictated by a uploadable optimal descend velocity of the RUAV. This descend velocity determines when the RUAV should climb onto the glide slope. Heave position and rate is fed directly into the heave controller to close-track the trajectory of the planned flight path.

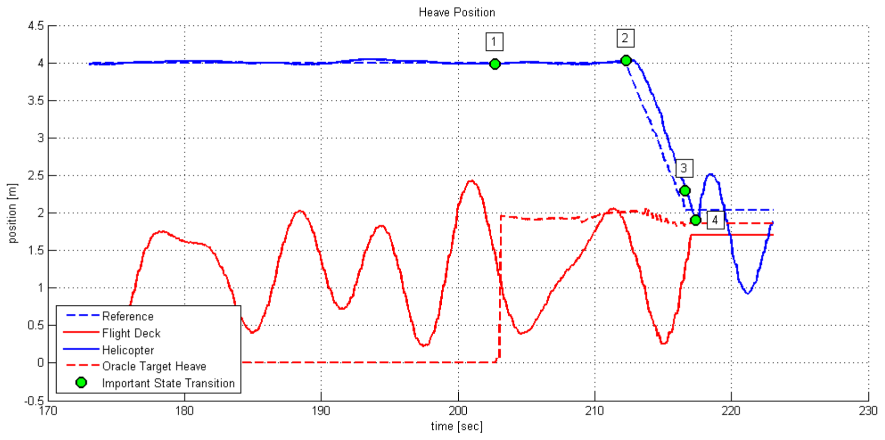
Stellenbosch University http://scholar.sun.ac.za



(a) Down Position



(b) Descend Velocity



(c) Detailed Heave Plot

Figure 8.9 – HIL Flighttest 3 : Flight Log Continue

Commit to Land [3]

At this instance the the Perching state transitions transitions to the Forced landing mode, where pure heave rate regulation is enforced to commit the RUAV for a touchdown on the flight deck.

Touchdown [4]

The accelerometer spike close to the vicinity of the flight deck triggers the transition of the state machine to the Touchdown low-level state. All control is dumped and the RUAV is shutdown in this state.

Stellenbosch University http://scholar.sun.ac.za

Chapter 9

Flight Tests

The two critical flight tests that took place was on the 25th of October 2012, §9.1,and the 5th of November 2012, §9.2. To eliminate variables, these flight tests were conducted on a stationary flight deck to validate the sensor characteristics of the VCAN node. As a concrete reference, the EKF estimator of the RUAV will be initialized exactly in the middle of the flight deck pattern that will lock this position of the flight deck at zero.

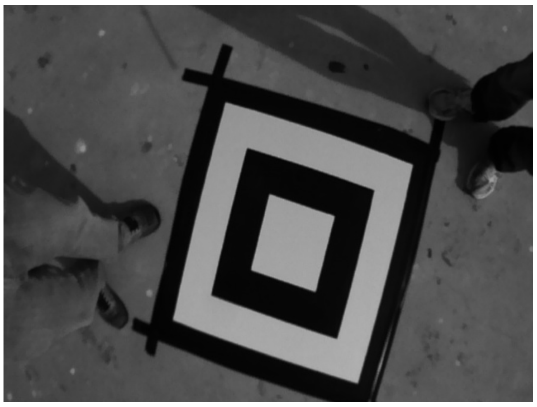


Figure 9.1 – Virtual Hover Lighting Calibration

The general conduct of each test is outlined below. The RUAV is first held in an assisted virtual hover above the flight deck pattern to configure the mt9v032 sensor of the VCAN node for optimal exposure and analogue gain levels using the feedback provided by the automatic filter. Notice the very distinct contrasts of the flight deck pattern as captured in Figure 9.1 during the virtual hover.

Stellenbosch University http://scholar.sun.ac.za

The RUAV is then positioned in the middle of the flight deck before the EKF estimator is initialized. The inner-loop controller gains and navigation parameters are uploaded to the OBC. After the motor of the RUAV is started, a manual takeoff from the safety pilot brings it to a fairly still hover. The autopilot of the RUAV is engaged by the safety pilot on request by the GS operator.

Stellenbosch University http://scholar.sun.ac.za

9.1 Flight Test 1 : 25-10-2012

This flight test was performed on the 25th of October 2012, at the Bosch-en-Dal Sport Fields in Stellenbosch. Wind conditions were not ideal as gusts of 6 m/s were recorded. The latter wind disturbances propagates through the inner-loop controllers of the RUAV as process noise. A flight deck pattern size of dimension 48 cm x 48 cm was used and a safety hover height of 5 m was enforced as the Perching hover height. This translates to a hover height ratio of $\frac{5}{0.48} = 10.416$. This hover height ratio was chosen as a conservative measure to acquire VL on the flight deck pattern. The theoretical expected noise levels that correlate to this node is given in Table 9.1.

σ_{north}	σ_{east}	σ_{down}	σ_{roll}	σ_{pitch}	σ_{yaw}
16.02 mm	25.18 mm	317.32 mm	14.05 deg	11.34 deg	2.564 deg

Table 9.1 – Flight Test 1 : Expected Sensor Noise

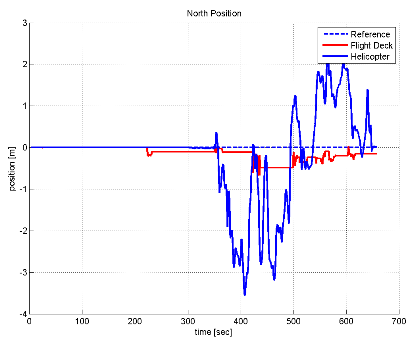
9.1.1 Existing Autopilot

The purpose of this flight test was to validate the performance of the VCAN node. As such, the RUAV was only stepped into the existing autopilot to stabilize the RUAV in an inertial position hold. Various vertical, longitudinal and lateral steps were issued to partially occlude the flight deck pattern from the view of the VCAN node. The flight log of the onboard kinematic estimator is shown in Figure 9.2. Notice that the noise levels of the flight test correlate well with the those proposed in Table 9.1. The flat regions introduced within the flight deck estimates are where VL was lock lost on the flight deck pattern. The precision in the flight deck state estimates also increases as the RUAV comes closer to the flight deck pattern. A manual landing by the safety pilot was commenced.

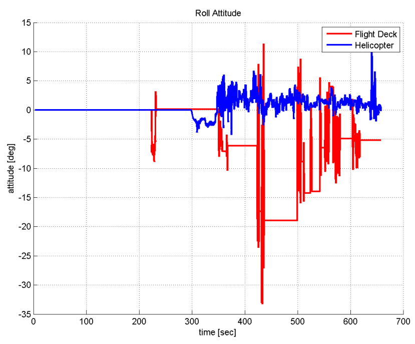
9.1.2 Perching Landing Controller

The purpose of this flight test was to validate the performance of the Perching landing controller. Snapshots taken during the envelope of this flight test is depicted in Figure 9.3. A smooth transition from the inertial position hold of the autopilot to the Perching landing controller was seen. A joint lateral and longitudinal step of 2 m was issued at a heave of 5 m to drift away from the flight deck pattern. The VCAN node flight deck information is used on the position lock command that returns the RUAV directly above the flight deck. A forced landing was authorized at a constant descend rate of 0.5 m/s.

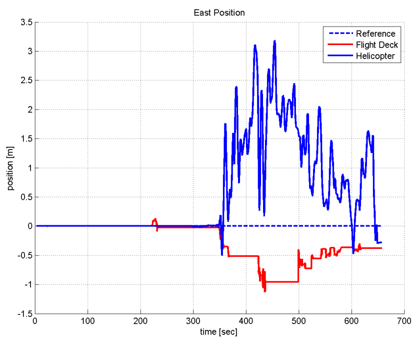
Stellenbosch University http://scholar.sun.ac.za



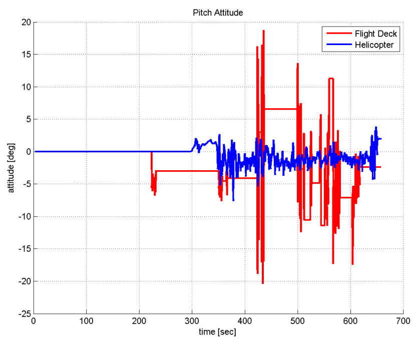
(a) North Position



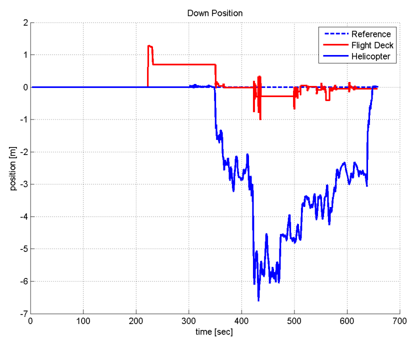
(b) Roll Attitude



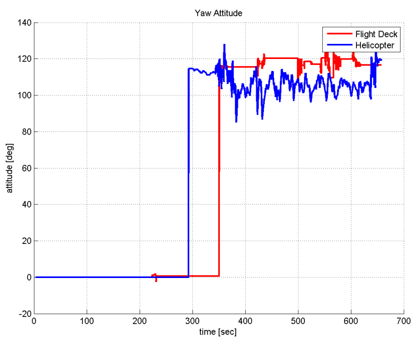
(c) East Position



(d) Pitch Attitude



(e) Down Position

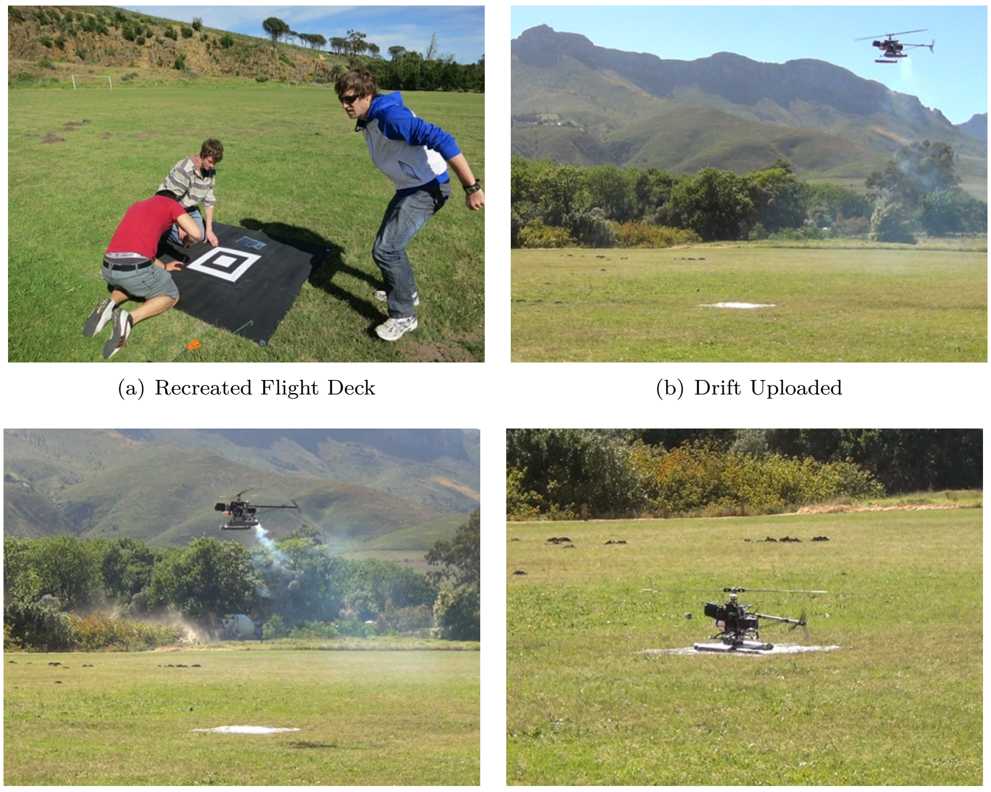


(f) Yaw Attitude

Figure 9.2 – Estimator Flight Log

Due to human error, an Oracle landing was authorized and since the predictor was not initialized for the stationary flight deck, the applicable LT was zero. This translates to a descend rate of infinity, but fortunately the inner velocity loop of the heave controller clamps a velocity reference at 1.5 m/s. Despite this error, the RUAV performed a hard, but not fatal landing. The flight log is shown in Figure 9.4

Stellenbosch University http://scholar.sun.ac.za



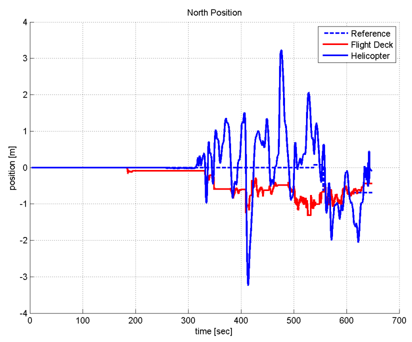
(a) Recreated Flight Deck (b) Drift Uploaded

(c) Position Lock Engaged (d) Touchdown

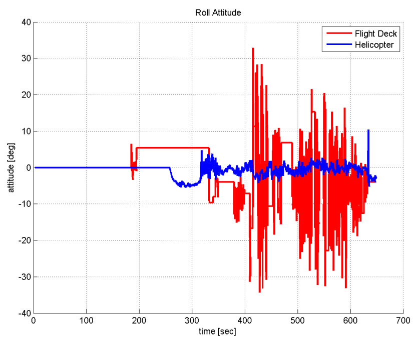
Figure 9.3 – Snapshots of Flight Test

The flight deck estimates during hover regulation for this flight test is utilized to correct the mounting misalignment of the VCAN node relative to the body axis of the RUAV. These corrections were made prior to §9.2 and §9.3.

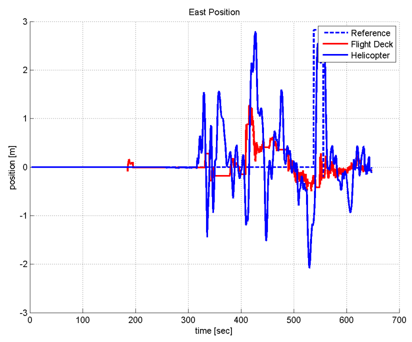
Stellenbosch University http://scholar.sun.ac.za



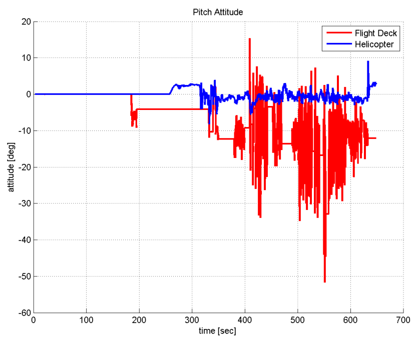
(a) North Position



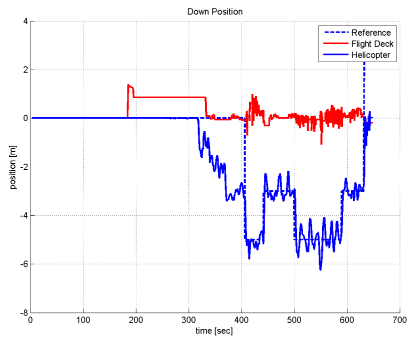
(b) Roll Attitude



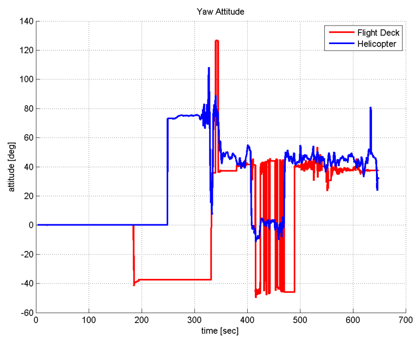
(c) East Position



(d) Pitch Attitude



(e) Down Position



(f) Yaw Attitude

Figure 9.4 – Estimator Flight Log

Stellenbosch University http://scholar.sun.ac.za

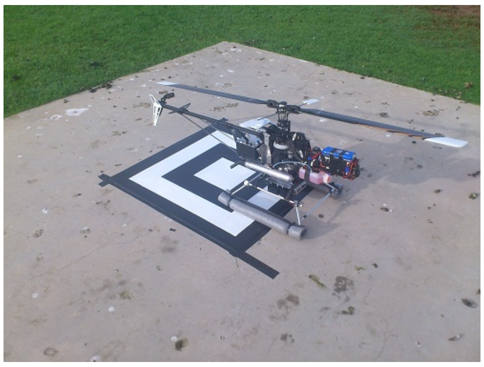
9.2 Flight Test 2 : 5-11-2012

This flight test was performed on the 5th of November 2012, on the Helderberg Radio Frequency hobby club in Somerset West. Wind conditions were very ideal during the entire envelope of this test. From the previous flight log, it was decided to make the following amendments to increase the accuracy of the inertial estimates of the flight deck:

- Increase the flight deck geometric pattern to 78 cm x 78 cm. This pattern size would not exceed a comfortable assisted lift height during the lighting calibration phase in virtual hover.
- Lower the perching hover height to 3 m.
- A joint lateral and longitudinal step of 3 m were uploaded during the drift phase.

The Gumstix COM was toggled to log aerial photos at 1 Hz as show in Figure 9.6. Figure 9.5 depicts snapshot photos taken during the envelope of the flight test. Figure 9.7 shows the flight log of the estimator data for both the RUAV and the flight deck. Due to the amendments discussed earlier, the accuracy of the VCAN node increased as the hover height ratio, §4.4, dropped from $\frac{5000}{480} = 10.42$ to $\frac{3000}{780} = 3.84$. The noise levels correlate very well to the predicted theoretical models. Notice that the roll and pitch angle variances would be tolerable for the prediction algorithm. These noise levels imply that the motion prediction could indeed be expanded to monitor as well as predict the roll and pitch angles of the flight deck.

Stellenbosch University http://scholar.sun.ac.za



(a) RUAV next to Flight Deck Pattern

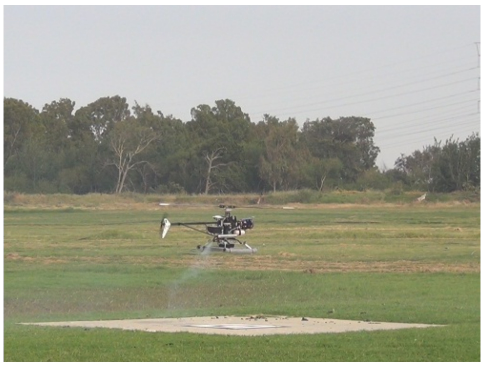
(b) Author on Ground Station

(c) Autopilot Position Hold

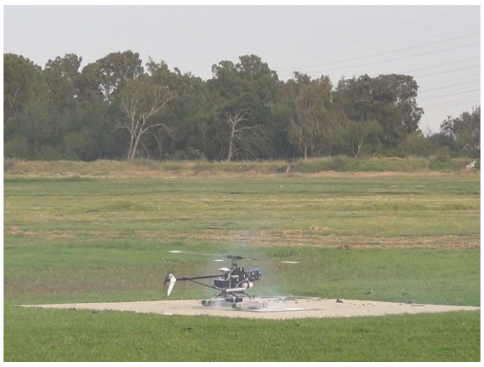
(d) Drift Uploaded

(e) Position Lock Engaged

(f) Authorized Forced Landing Descent



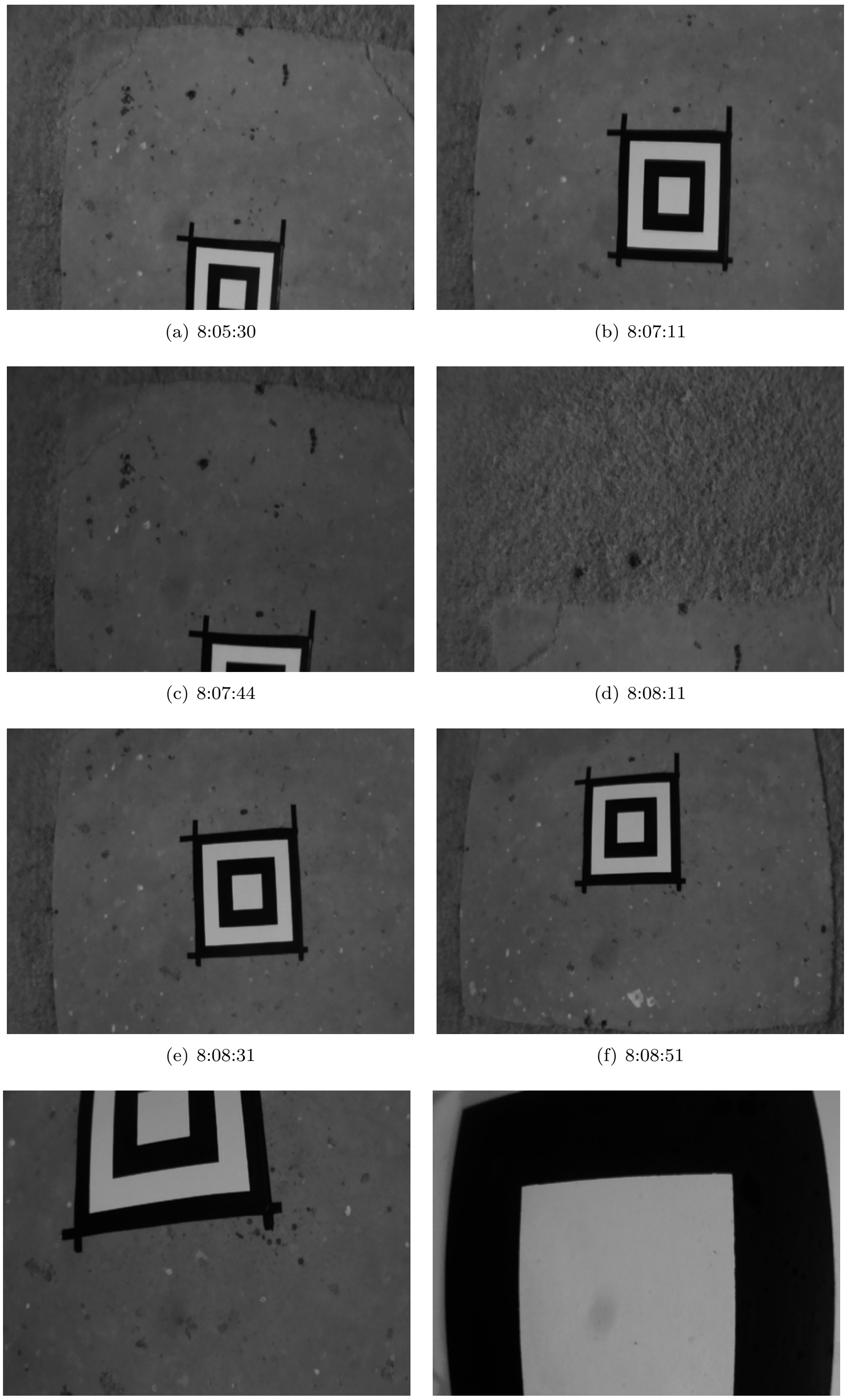
(g) Ground Effect Cushion



(h) Safe Touchdown

Figure 9.5 – Snapshots of Flight Test

Stellenbosch University http://scholar.sun.ac.za



(a) 8:05:30

(b) 8:07:11

(c) 8:07:44

(d) 8:08:11

(e) 8:08:31

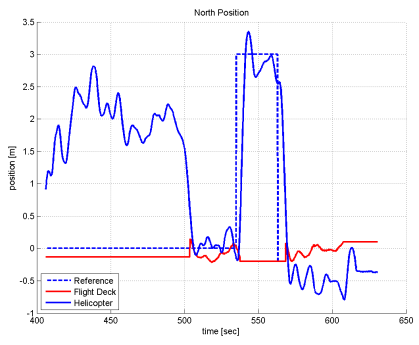
(f) 8:08:51

(g) 8:08:55

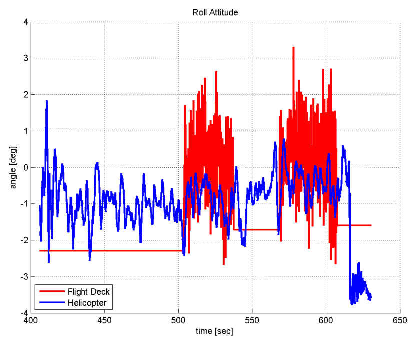
(h) 8:08:59

Figure 9.6 – Aerial Photos

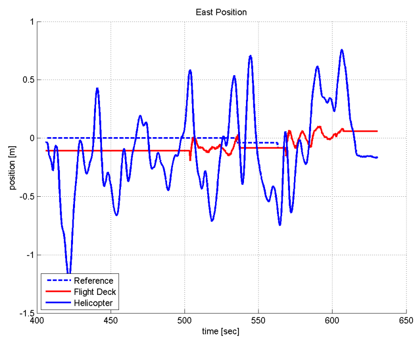
Stellenbosch University http://scholar.sun.ac.za



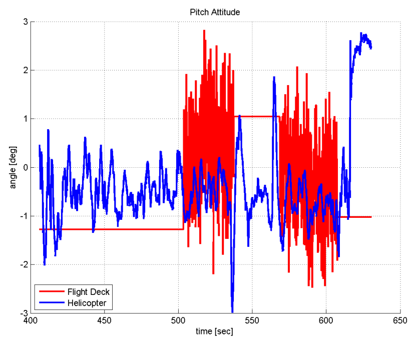
(a) North Position



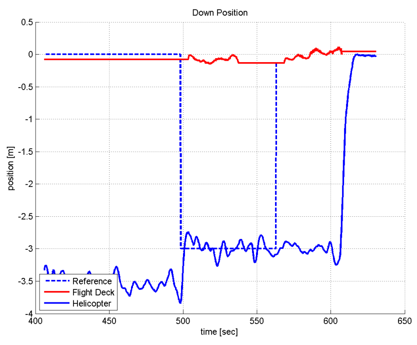
(b) Roll Attitude



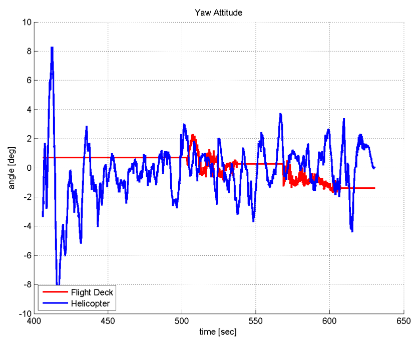
(c) East Position



(d) Pitch Attitude



(e) Down Position



(f) Yaw Attitude

Figure 9.7 – Estimator Flight Log

Stellenbosch University http://scholar.sun.ac.za

9.3 Flight Test 3 : 3-12-2012

This flight test was performed on the 3rd of December 2012, at the Engineering Faculty in Stellenbosch. The purpose of this test was to validate the prediction capabilities of the VCAN node. Although all the RUAV avionics were armed on the day of the test, this was strictly speaking, not an actual flight test since the RUAV's motor was not live. Due to weather conditions, it was decided to suspend the RUAV with a forklift over the pneumatic platform and perform the prediction tests as illustrated in Figure 9.8. A HD Wing 1280x720p 30fps 5MP CMOS pocket camera was mounted to the under carriage of the RUAV to provide video footage from a bird's eye view. Two dedicated Mac Afric 70L 6.5HP compressors were used to meet the system's pneumatic demands.

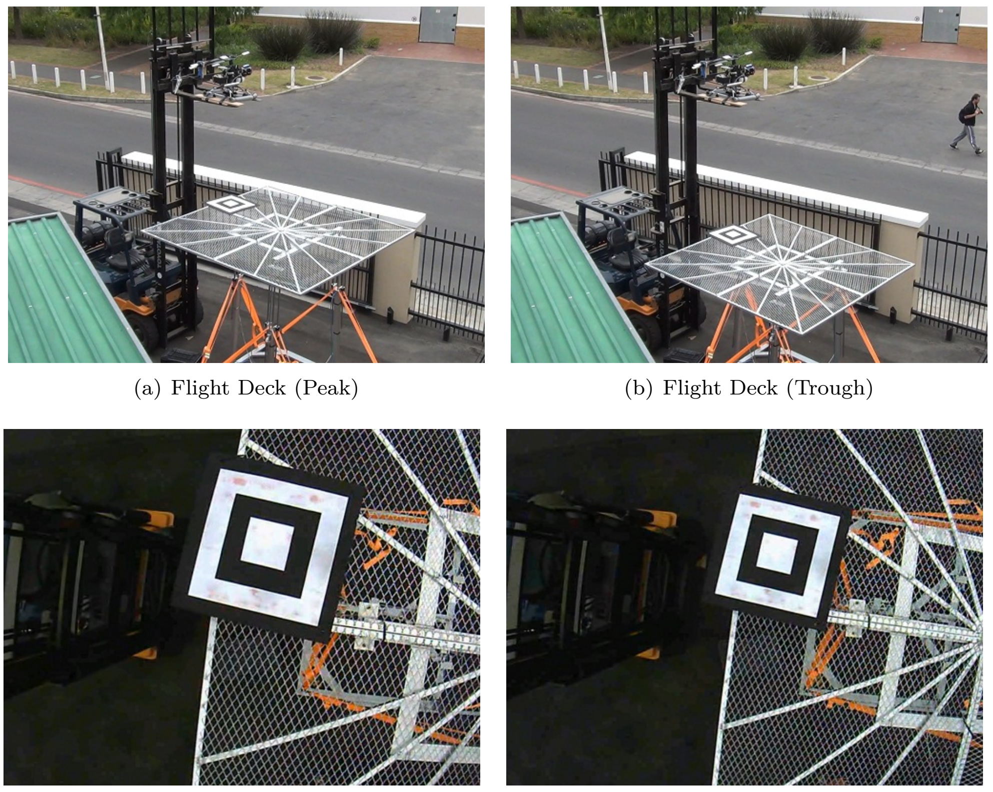
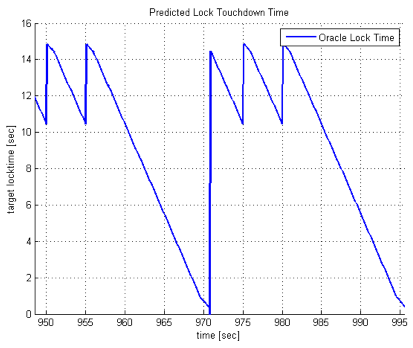

(a) Flight Deck (Peak)
(b) Flight Deck (Trough)
(c) Bird's eye View (Peak)
(d) Bird's eye View (Trough)

Figure 9.8 – Prediction Tests

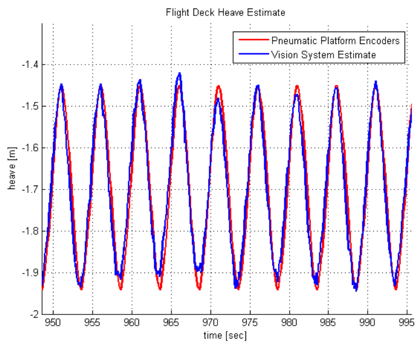
The avionics onboard the RUAV was armed as for an actual flight test. The RUAV outer-loop controller was commanded to transition into the Perching state and test the Oracle landing mode. The vision system was allowed to build up a regression history of the flight deck before the prediction algorithm was initialized. At this stage, the furthermost safe

Stellenbosch University http://scholar.sun.ac.za

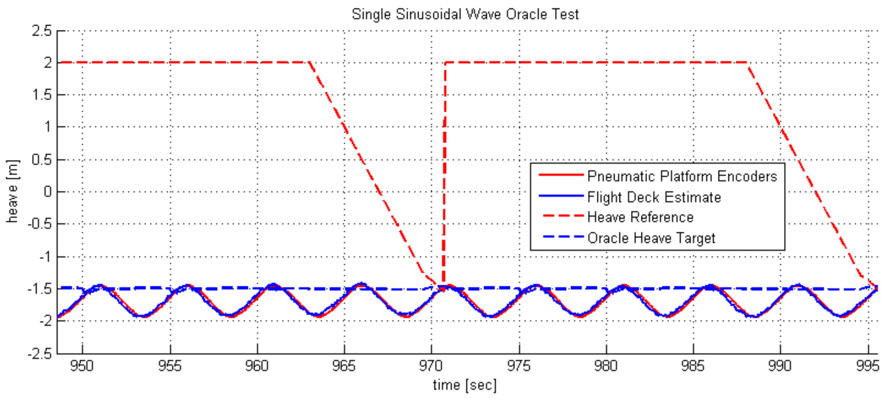
landing opportunity LT was streamed down to the GS. When the authorization for landing commanded was issued, the vision system latched onto the current LT and followed it through until the virtual touchdown. The latter LT was used to construct a descend glide slope. This process was repeated a number of iterations to gather virtual landing data to validate the Oracle landing mode. After the authorization for landing command was given, the deck down target and LT variables were closely monitored. The GS operator was also notified when the heave reference climbed onto the glide slope and when the transition to the Forced landing mode occurred.



(a) Oracle Locktime



(b) Flight Deck Heave Estimate



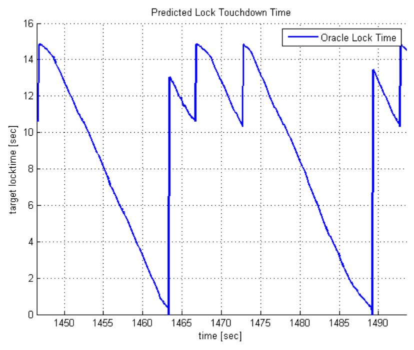
(c) Oracle Landing Controller Logic

Figure 9.9 – Single Sinusoidal Wave Oracle Test

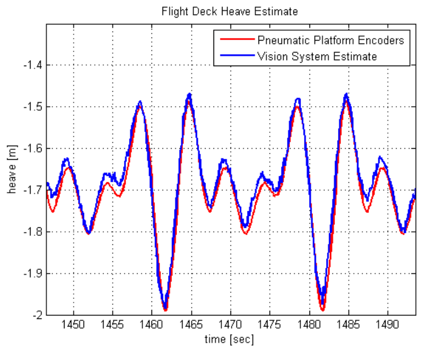
The EKF estimator of the RUAV was initialized above the flight deck, hence the negative bias on the heave estimates in Figures 9.9 and 9.10. For the first trial, the pneumatic flight deck was excited with a single sinusoidal wave of 0.25 m amplitude and 0.2 Hz frequency. The flight log for this wave is given in Figure 9.9. For the second trial, the pneumatic flight deck was excited with a multiple sinusoidal wave of 0.25 m amplitude and 0.2 Hz, 0.15 Hz

Stellenbosch University http://scholar.sun.ac.za

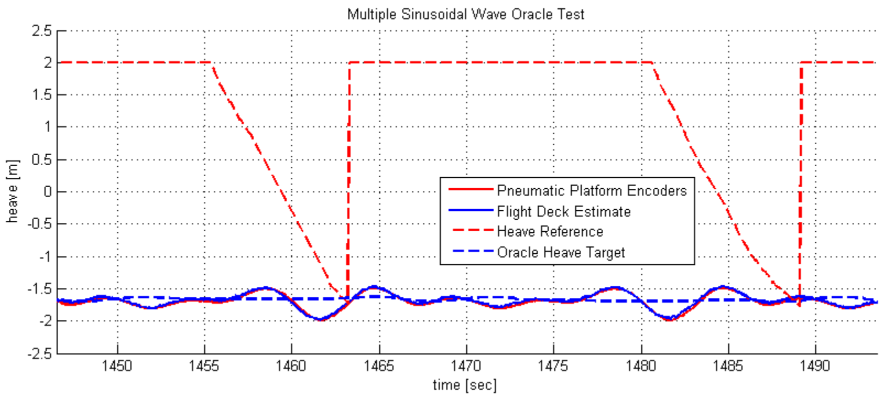
and 0.1 Hz frequencies. The flight log for this wave is given in Figure 9.10. The perch height was defined at a 2 m altitude and the optimal descend velocity was set to 0.5 m/s.



(a) Oracle Locktime



(b) Flight Deck Heave Estimate



(c) Oracle Landing Controller Logic

Figure 9.10 – Multiple Sinusoidal Wave Oracle Test

Immediately after the authorization to land command was given, the motion prediction algorithm latches onto the furthermost safe landing opportunity and follows it through until touchdown. The predicted heave for the flight deck that correlates to the predicted safe landing point is also streamed down. After the first virtual touchdown, authorization to land was denied that commanded the RUAV to return to the safe perching hover height at 2 m. Another Oracle landing was attempted to re-illustrate this logic of the landing controller for both trials.

The hover height references for Figures 9.9 and 9.10 was injected directly into the heave controller. It can thus be extrapolated from these trials that the Oracle landing mode will indeed be successful. The vision system is capable of supplying accurate inertial state

Stellenbosch University http://scholar.sun.ac.za

information of the flight deck to the motion prediction algorithm. This section has illustrated how the predictions will be injected into the landing controller that plans and follows through on a controlled descend.

Stellenbosch University http://scholar.sun.ac.za

Chapter 10

Conclusion and Recommendations

10.1 Conclusion

The problem statement of the autonomous landing of a helicopter on a moving flight deck was comprehensively presented in the introduction to this thesis. A thorough literature review was presented on critical aspects such as computer vision and motion prediction. The concepts of pattern recognition and pose estimation were well formulated, optimized and implemented on the VCAN board to provide inertial information of the flight deck. The flight deck estimate noise levels from actual flight log data compared well to the constructed theoretical models. The Forced landing mode was tested in close conjunction with the VCAN node and subsequently two successful, autonomous vision-aided landings were performed.

The motion prediction algorithm was well formulated and benchmarked against actual frigate data. The forecast capabilities of the VCAN node were tested in §9.3. The Oracle landing mode was thoroughly tested in HIL simulations that deemed landing on a heaving platform to be viable for actual flight tests. The following contributions to the ESL research repository have been made by the author:

- Native code development environment on the Gumstix COM.
- Development of a vision system that provides fast, high accuracy, relative state information.
- Development of pattern recognition, pose estimation and motion prediction codes.
- Development of the VCAN board that connects the Gumstix COM to the avionics CAN bus.

Stellenbosch University http://scholar.sun.ac.za

- Pioneer in using Gumstix COM during flight.
- Pioneer in vision assisted autonomous landing of an UAV.

All of the components of the project reached their outcomes, except the actual flight test to validate the performance of the Oracle landing mode, as depicted in Figure 10.1. This landing mode was, however, thoroughly tested in HIL simulations (§8.2) and the previous flight test logs (Chapter 9) proved the vision assisted RUAV creditable for a Oracle landing attempt. The HIL simulation data correlates very well to the above mentioned flight logs.

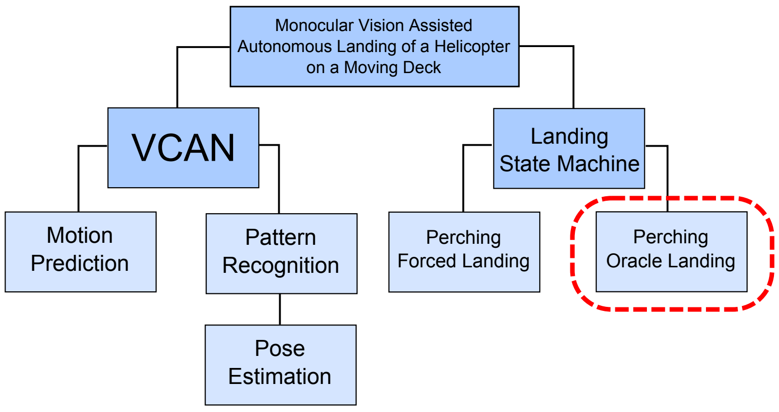


Figure 10.1 – Project Outcomes

There are two major reasons that led up to this disappointment of not completing a actual Oracle landing:

- The pneumatic 3DOF platform that needs to simulate the flight deck, was only fully operational two weeks prior to the deadline of this thesis. The weather conditions prior to the deadline did not allow for another flight test.
- The longitudinal and lateral inner-loop controllers of the RUAV are very slow to reject wind disturbances and pose serious safety issues to land on a heaving platform during windy conditions. The RUAV control system, in particular the longitudinal and lateral controllers, require an update in both architecture and gains to provide the required stability and robustness for a safe autonomous landing on a moving platform. Such an effort was however outside the scope of this thesis since ample system identification flight tests were required.

Stellenbosch University http://scholar.sun.ac.za

10.2 Recommendations

The following recommendations are made:

10.2.1 Vision System

- It is highly recommended to compile a custom Angstrom desktop image kernel for the Gumstix COM as opposed to the console image kernel used in this project. This would make native code development much easier and also allow for the option to display the streamed footage from the connected camera.
- The most difficult part of the vision system is the initial lighting calibration test to obtain VL on the flight deck pattern. The GS operator must familiarize himself with the effects of different exposure levels and analogue gains of the camera. It is highly recommended to automatically populate a 3D space described by exposure level, analogue gain and threshold value of where VL is obtained, as shown in Figure 10.2. The centroid of the largest cloud where the flight deck pattern was detected, is the optimal vision parameters.

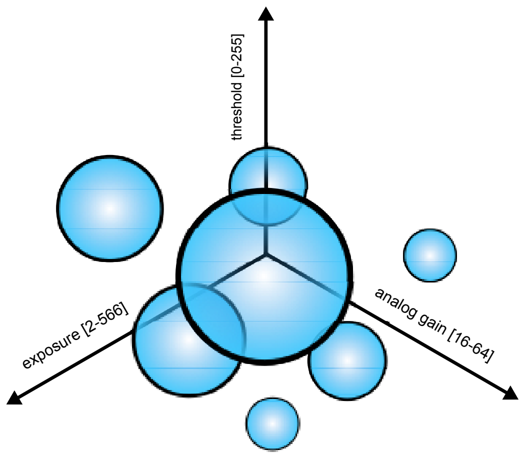


Figure 10.2 – Visual Lock Cloud

10.2.2 Control System

- Numerous system identification flight tests must be scheduled for a better representation of the X-Cell Fury Expert dynamics. Actual flight test data concludes that the X-Cell Fury Expert responds considerably faster in reality than the dynamics described within the non-linear HIL simulation environment.

Stellenbosch University http://scholar.sun.ac.za

- Subsequently it is then recommended to tighten the inner-loop control gains. The low control gains of acrobatic X-Cell helicopter is very conservative as seen during hover regulation in the two flight tests of Chapter 9.

- It is highly recommended to follow through on the update for the kinematic estimator onboard the RUAV as described in §6.3.

- Should the control system be ported onto the Gumstix COM, it is highly recommended to implement more modern control strategies such as LQR or MPC.

10.2.3 VCAN

- The VCAN board was developed as a short-term solution to provide a data link between the Gumstix COM and the CAN bus. Eight pins on the 40-GPIO header located on the Summit expansion board, can be used as a SPI port to interface directly with the OBC. Utilizing the expansion board provides a means to terminal into the Gumstix COM at any time. The expansion board provides numerous USB interfaces to support a keyboard and mouse. If desktop kernel images are loaded onto the Gumstix COM, the HDMI port can be used to stream to a monitor. Such a configuration results in creating a setup similar to a desktop PC.

10.2.4 Gumstix

- It is recommended to source an expansion board for the Gumstix COM, similar to the Overo Summit board, that supports networking capabilities. Cross-platform development lost its appeal as it was very tedious to transfer files between the Gumstix and the development PC. Subsequently, all the code development had to be performed on the actual Gumstix COM via terminal, using Tera-Term.

- The CaspaPX camera board used in this project is the prescribed, Gumstix COM approved camera that interfaces with I2C. If a expansion board were to be used, the available USB port could support a variety of USB cameras.

- The CAN protocol can be implemented on the GPIO header of the Summit expansion board. By doing this, the VCAN board is totally omitted.

Stellenbosch University http://scholar.sun.ac.za

Appendix A

Collision Dynamics

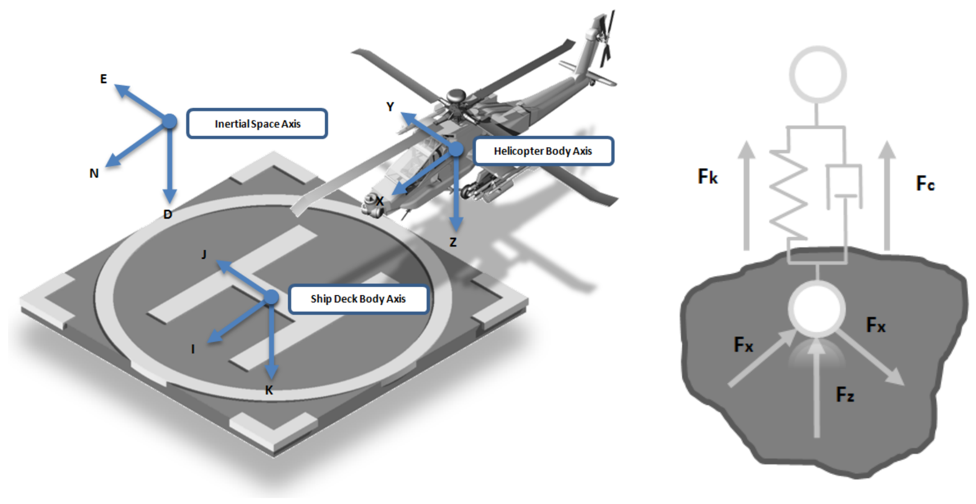
This chapter describes the fundamental collision forces and coupled moments that two colliding six-degree-of-freedom bodies experiences upon impact. This thesis stems from the automatic takeoff and landing division and as such, considerable emphasis is placed on this chapter since one must be able to simulate, for example, how the flight deck of the ship can potentially knock the helicopter out of hover in mid-flight. As such, the full derivation of the equations that govern the dynamic interaction between the helicopter skids and the flight deck of a ship will be provided.

The following assumptions were made:

1. The flight deck of the ship will be modelled as a plane.
2. The skids of the helicopter will be modelled as four separate contact points which are offset from the helicopter's centre of gravity.
3. Each contact point's kinematics are modelled as a spring and damper system whose normal forces and viscous friction forces act perpendicularly and co-planar with the flight deck respectively.
4. Applying the conservation of momentum the change of velocity of the ship after the impact with the helicopter could be argued to be negligible due to the ship's sheer size.

Figure A.1 shows the three independent axes defined by NED as the quasi inertial space axis, XYZ the body axis of the helicopter and IJK the body axis of the flight deck. The position of a contact point, defined from the CG of the helicopter $\vec{Skids}_{XYZ}$, can be relocated to their inertial space vector, $\vec{Skids}_{NED}$ and finally into the flight deck axis, $\vec{Skids}_{IJK}$. $\left(DCM^T_{Heli}\right)$

Stellenbosch University http://scholar.sun.ac.za



(a) Axes (b) Skid Modelling

Figure A.1 – Collision References

is the direction cosine matrix for a transformation from the inertial coordinate system to the helicopter body axis. $\left(DCM^T_{Deck}\right)$ is the direction cosine matrix for a transformation from the inertial coordinate system to the flight deck body axis.

$$\vec{Skids}_{NED} = \left(DCM^T_{Heli}\right)\vec{Skids}_{XYZ} \tag{A.0.1}$$

$$\vec{Skids}_{IJK} = \left(DCM^T_{Deck}\right)\left(\vec{Skids}_{NED} - \vec{Deck}_{NED}\right) \tag{A.0.2}$$

The respective velocity vectors for each contact point can be represented by $\dot{\vec{Skids}}_{NED}$ in the inertial coordinate system and be relocated into the flight deck axis, $\dot{\vec{Skids}}_{IJK}$. Note that $\vec{PQR}_{XYZ}$ represents the Euler angle rates of the helicopter body axis with reference to the inertial coordinate system.

$$\dot{\vec{Skids}}_{NED} = \left(DCM^T_{Heli}\right)\left(\vec{PQR}_{XYZ} \times \vec{Skids}_{XYZ}\right) + \dot{\vec{Heli}}_{NED} \tag{A.0.3}$$

$$\dot{\vec{Skids}}_{IJK} = \left(DCM^T_{Deck}\right)\left(\dot{\vec{Skids}}_{NED} - \dot{\vec{Deck}}_{NED}\right) - \vec{PQR}_{IJK} \times \vec{Skids}_{IJK} \tag{A.0.4}$$

$$F_K = \begin{cases} -Ks \cdot (Ls + K) - Kv \cdot (\dot{K}) & \text{if} Ls + K > 0, \\ 0 & \text{if} Ls + K <= 0. \end{cases} \tag{A.0.5}$$

Stellenbosch University http://scholar.sun.ac.za

$$\begin{bmatrix} F_I \\ F_J \end{bmatrix} = \begin{bmatrix} -Bd \cdot F_K \cdot sign\left(\dot{I}\right) \\ -Bd \cdot F_K \cdot sign\left(\dot{J}\right) \end{bmatrix} \tag{A.0.6}$$

$$\vec{F}_{XYZ} = \left(DCM_{Heli}\right)\left(DCM^T_{Deck}\right)\vec{F}_{IJK} \tag{A.0.7}$$

$$\vec{M}_{XYZ} = Sk\vec{i}ds_{IJK} \times \vec{F}_{XYZ} \tag{A.0.8}$$

Stellenbosch University http://scholar.sun.ac.za

Appendix B

VCAN

Description	Package	Bits	Range	Resolution
Identifier	0	15-0	0xAA55	-
Heli North	1	15-0	+- 15 m	0.4578 mm
Heli East	2	15-0	+- 15 m	0.4578 mm
Heli Down	3	15-0	+- 15 m	0.4578 mm
Heli Roll	4	15-0	+- 90 deg	0.0027 deg
Heli Pitch	5	15-0	+- 90 deg	0.0027 deg
Heli Yaw	6	15-0	+- 90 deg	0.0027 deg

Table B.1 – GX TX NORMAL [CAN EID : 0x0B810108/0x0B820108]

Description	Package	Bits	Range	Resolution
Identifier	0	15-0	0xAA55	-
Deck North	1	15-0	+- 15 m	0.4578 mm
Deck East	2	15-0	+- 15 m	0.4578 mm
Deck Down	3	15-0	+- 15 m	0.4578 mm
Deck Roll	4	15-0	+- 90 deg	0.0027 deg
Deck Pitch	5	15-0	+- 90 deg	0.0027 deg
Deck Yaw [Target Down]	6	15-0	+- 90 deg [15m]	0.0027 deg [0.4578 mm]
Lock time	7	15-8	0-10 sec	0.0392 sec
Visual Lock	7	7	0/1	-
Logging Photo	7	6	0/1	-
Synchronous Delay	7	5	0/1	-
Parameters Configured	7	4	0/1	-

Table B.2 – GX RX NORMAL [CAN EID : 0x02810601/0x02820601]

Flooding the CAN Bus with data from the CAN-Vision board can result in a RX package fault at the Servo board command. With the loss of the Servo board, the Helicopter will be rendered uncontrollable. To upload a settings message for the Gumstix from the GS,

Stellenbosch University http://scholar.sun.ac.za

the same buffers will be used but with secondary identifier to validate that it is indeed a configuration message.

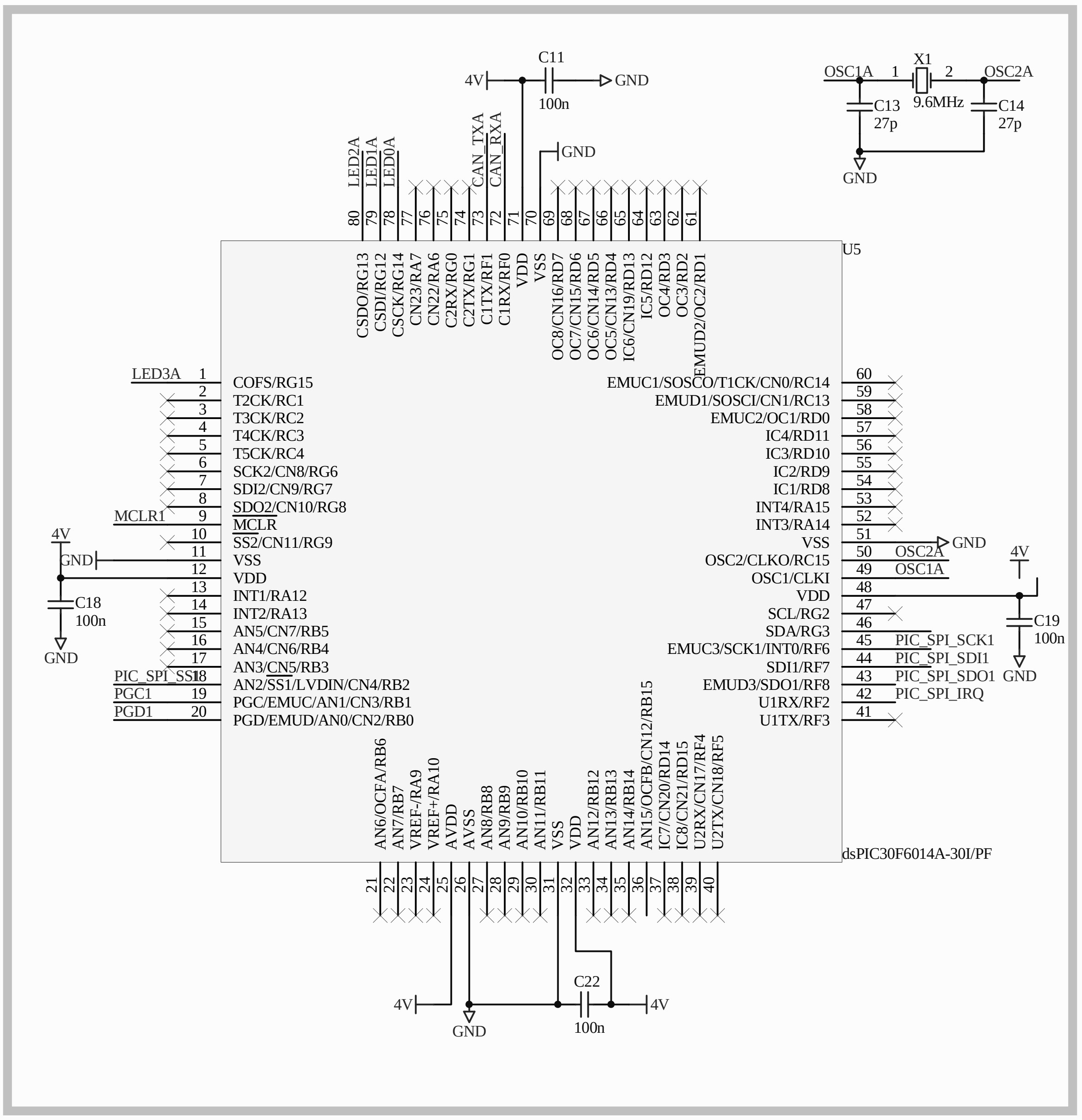
Description	**Package**	**Bits**	**Range**	**Resolution**
Primary Identifier	0	15-0	0xAA55	-
Secondary Identifier	1	15-0	0xBB55	-
Exposure	2	15-8	2-566	2.2117
Analog Gain	2	7-0	16-64	0.1882
Contrast Enhancement	3	15-8	100-500	1.5686
Smallest Quadrant Block size	3	7-0	0-500 mm	2 mm
Photos Logging	4	15	0/1	-
Lowlight Compensation	4	14	0/1	-
Fire Adaptive Filter	4	13	0/1	-
Return Relative Measurements	4	12	0/1	-

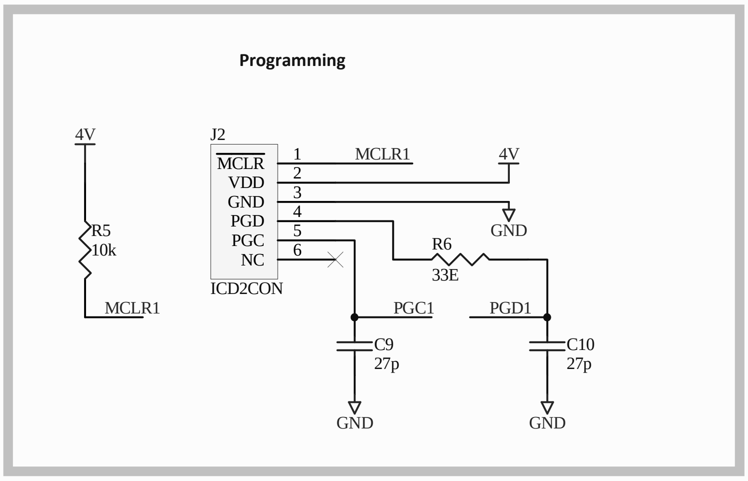
Table B.3 – GX TX SETTINGS [CAN EID : 0x0B810108/0x0B820108]

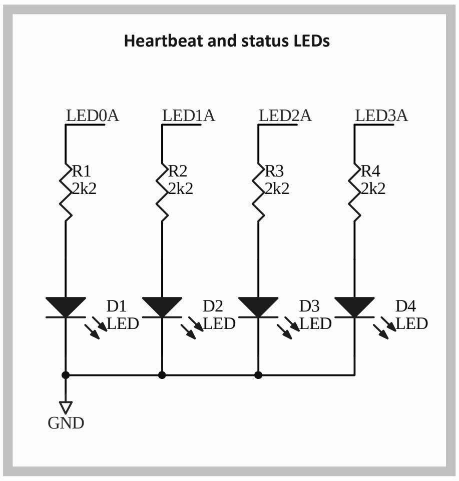
Description	**Package**	**Bits**	**Range**	**Resolution**
Primary Identifier	0	15-0	0xAA55	-
Secondary Identifier	1	15-0	0xBB55	-
Filter Converged Threshold	2	15-8	0-255	1
Filter Bandwidth	2	7-0	0-255	1
Lock time	7	15-8	0-10 sec	0.0392 sec
Visual Lock	7	7	0/1	-
Logging Photo	7	6	0/1	-
Synchronous Delay	7	5	0/1	-
Parameters Configured	7	4	0/1	-

Table B.4 – GX RX SETTINGS [CAN EID : 0x02810601/0x02820601]

Stellenbosch University http://scholar.sun.ac.za

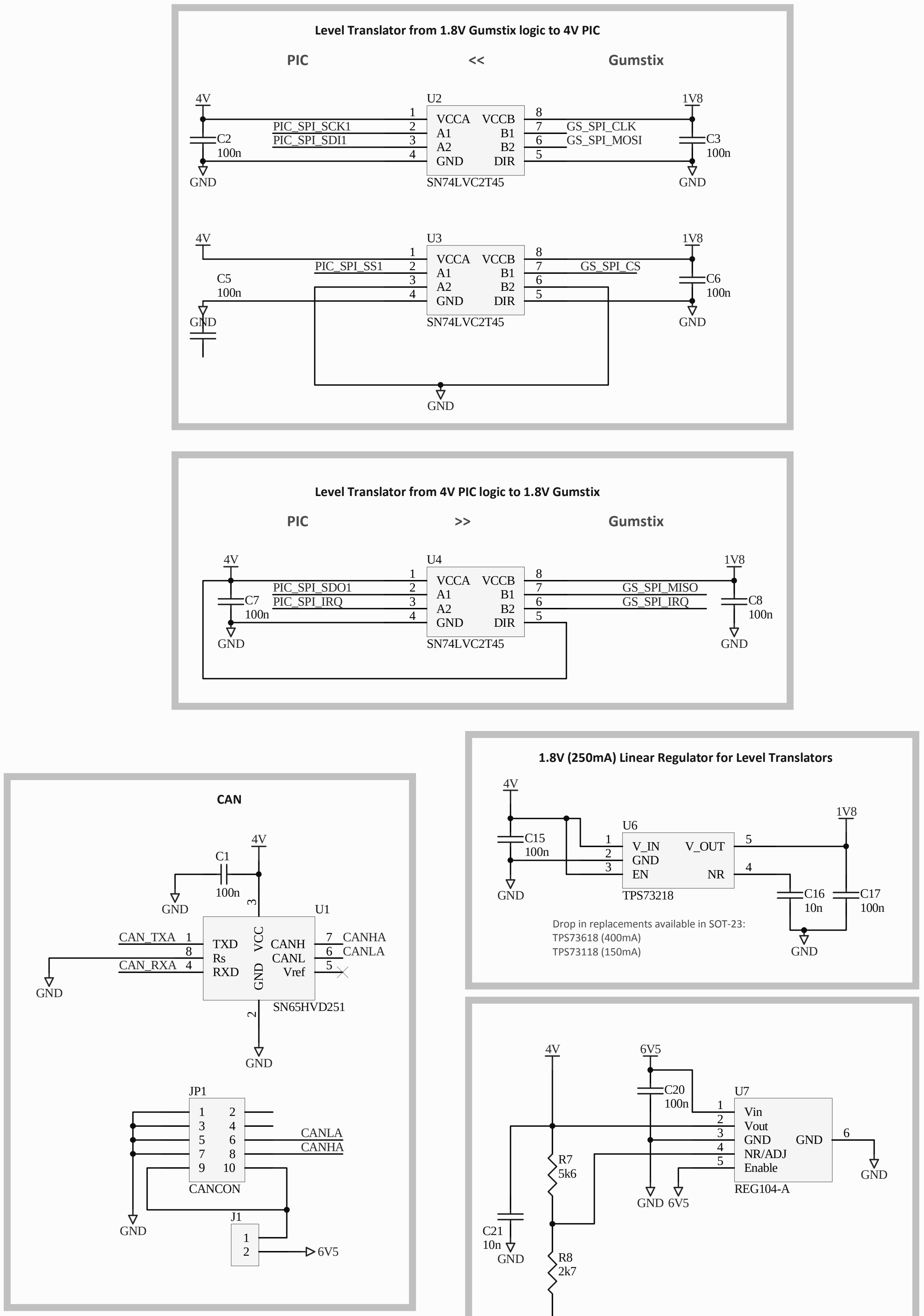






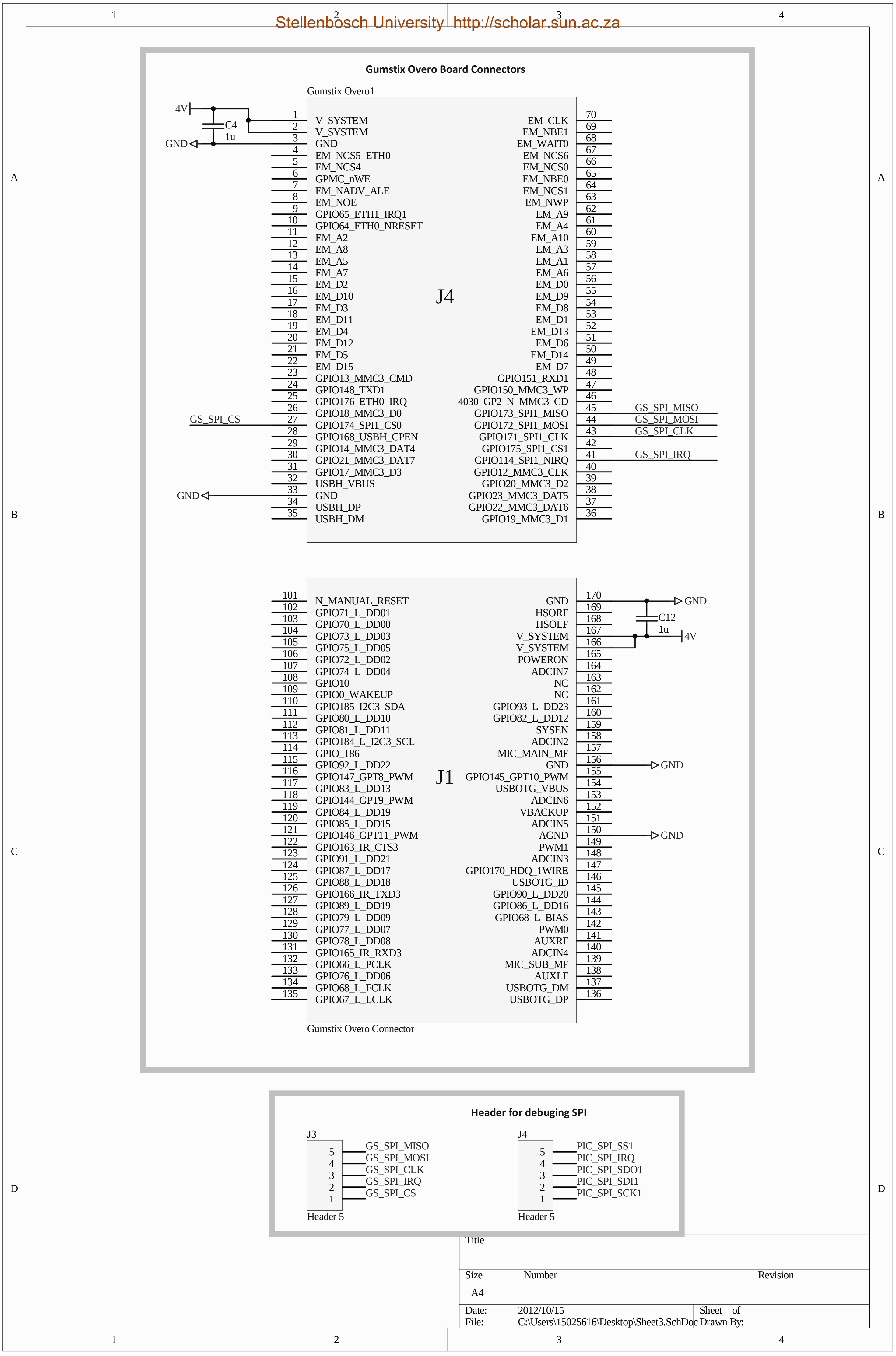
Title		
Size A4	Number	Revision
Date: 2012/10/15		Sheet of
File: C:\Users\15025616\Desktop\Sheet1.SchDoc		Drawn By:

Stellenbosch University http://scholar.sun.ac.za



Title		
Size A4	Number	Revision
Date:	2012/10/15	Sheet of
File:	C:\Users\15025616\Desktop\Sheet2.SchDoc	Drawn By:

Stellenbosch University http://scholar.sun.ac.za



Stellenbosch University http://scholar.sun.ac.za

Appendix C

Calibration

C.1 Checkerboard Pattern Calibration

The author can be seen in Figure C.1 during this checkerboard calibration.

C.2 VCAN Offset Calibration

During autonomous flight, the small angular and translation offsets from the avionics stack and the NovAtel receiver with respect to the CG of the RUAV is considered negligible, as seen in Figure 4.3. During the final envelope of the flight, far more precise measurements will be needed for a controlled landing. It was assumed that the VCAN node will inherit the RUAV's attitude as measured by the avionics stack. The position offset of the Novatel Receiver relative to the skids are automatically accounted for when the RUAV's estimator is initialized on the ground. The VCAN node is situated at [0.2,0,0.36] with respect to the Novatel Receiver [0,0,0], aligned to the RUAV body axis. The center of the landing gear is situated at [0.17,0,0.4] with respect to the Novatel Receiver. Should the landing gear not be taken into account, then the position estimate of the flight deck will be offset by the center of the landing gear. For control reference purposes, it was chosen to combine the VCAN and the landing gear center to a global offset of [-0.03,0,0.04]

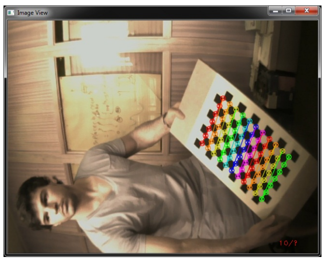
$$X = Z\frac{x - cx}{fx} = 1.5\frac{320 - 331.795}{543.913} = -0.0325 \tag{C.2.1}$$

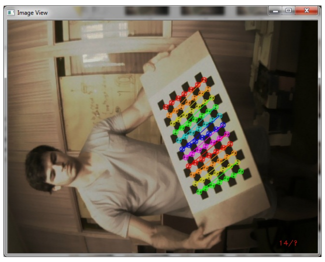
Equation C.2.1 also illustrates the principal x offset that translates to a position offset in the relative east measurement. This offset is however automatically accounted for in the null space solution. The misalignment of the lens in the y-direction is considered negligible.

Stellenbosch University http://scholar.sun.ac.za

Parameter	Calibrated Value
fx	5.43913e+002
fy	6.36865e+002
cx	3.31795e+002
cy	2.39237e+002
k1	-4.34530e-001
k2	2.36155e-001
p1	3.056710e-003
p2	-1.93250e-003

Table C.1 – Intrinsic and Distortion Values


(a) Frame 9


(b) Frame 26

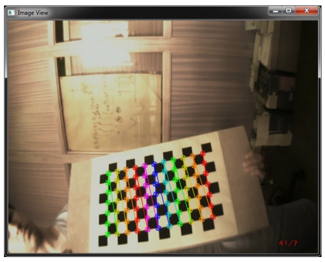

(c) Frame 41

Figure C.1 – Checkerbox Calibration

C.3 Virtual Noise Levels

Each of the latter lattice plots depicted in Chapter 4 are approximated analytically by the following surface fit equation:

$$Z(h,r) = Ae^{Bh} + Ce^{Dr} \tag{C.3.1}$$

where Z represents one standard deviation, h and r the hover height and drift radius ratios respectively. A global error over the entire envelope is given by:

$$\sum_{i=1}^{n} E^2 = \sum_{i=1}^{n} \left(Z(h,r) - Ae^{Bh} - Ce^{Dr}\right) \tag{C.3.2}$$

Each parameter can be adapted into the direction that minimizes the global error. The error partial derivatives are taken:

$$\frac{\delta E}{\delta A} = 2\sum_{i=1}^{n} \left(-Z(h,r)e^{Bh} + Ae^{2Bh} + Ce^{Bh+Dr}\right) \tag{C.3.3a}$$

$$\frac{\delta E}{\delta B} = 2\sum_{i=1}^{n} \left(-Z(h,r)ABe^{Bh} + A^2Be^{2Bh} + ABCe^{Bh+Dr}\right) \tag{C.3.3b}$$

Stellenbosch University http://scholar.sun.ac.za

$$\frac{\delta E}{\delta C} = 2\sum_{i=1}^{n}\left(-Z(h,r)e^{Bh} + Ae^{Bh+Dr} + Ce^{2Dr}\right) \tag{C.3.3c}$$

$$\frac{\delta E}{\delta D} = 2\sum_{i=1}^{n}\left(-Z(h,r)CDe^{Dr} + ADCe^{Bh+Dr} + C^2De^{2Dy}\right) \tag{C.3.3d}$$

After the residual error converged to a tolerable value, the noise levels were charaterized by the following analytical equations:

$$\sigma_{north[mm]} = 0.0175e^{0.6549h} + 1.7496e^{0.9301r} - 1.7833 \tag{C.3.4a}$$

$$\sigma_{east[mm]} = 0.0312e^{0.6749h} + 3.1227e^{0.7102r} - 3.1840 \tag{C.3.4b}$$

$$\sigma_{down[mm]} = 3.7503e^{0.3502h} + 15.0013e^{0.6603r} - 20.3233 \tag{C.3.4c}$$

$$\sigma_{roll[deg]} = 1.8986e^{0.2069h} + 2.4088e^{0.4680r} - 4.7437 \tag{C.3.4d}$$

$$\sigma_{pitch[deg]} = 4.2634e^{0.1281h} + 4.3932e^{0.3448r} - 9.2392 \tag{C.3.4e}$$

$$\sigma_{yaw[deg]} = 0.0821e^{0.3346h} + 0.1039e^{0.9906r} - 0.2186 \tag{C.3.4f}$$

Stellenbosch University http://scholar.sun.ac.za

Appendix D

Gumstix COM Kernel and Rootfs

A customized Angstrom Linux Kernel had to be compiled for this thesis that supported:

- inherits linux-omap3-caspapx-2.6.34 kernel
- native code developement (task-native-sdk)
- OpenCV libraries (opencv-dev)
- OpenCV DSP-Acceleration
- SPI device support (spidev-enable.patch)
- mt9v032 driver support (mt9v032-2.6.34.patch)

The on-line documentation that was used as close guidelines were [53], [50], [54] and [55]. Prior to installing the kernel and modules, the Gumtix COM SD card needs to be partitioned as shown in Figure D.1. Run the following console commands for the boot partition:

- cp MLO /media/boot
- cp u-boot.bin /media/boot
- cp uImage-2.6.34-r103-overo.bin /media/boot/uImage

Run the following console commands for the rootfs partition:

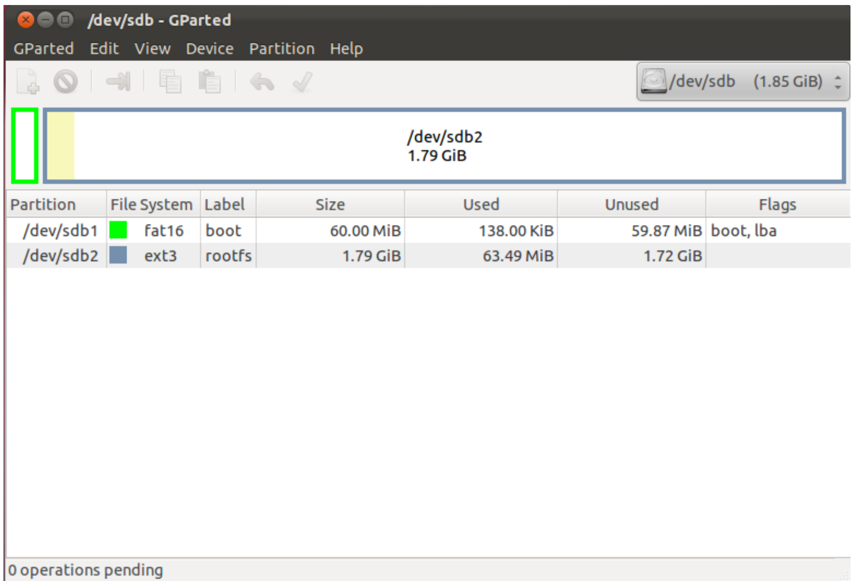
- tar -xjf Angstrom-caspa-console-image-glibc-ipk-2011.03-overo.rootfs.tar.bz2 -C /media/rootfs
- tar -xf modules-2.6.34-r103-overo.tgz -C /media/rootfs/

Stellenbosch University http://scholar.sun.ac.za

- cp uImage-2.6.34-r103-overo.bin /media/rootfs/boot/uImage

If the Gumsix COM will be accessed via Terminal, replace the following entry in /media/-rootfs/etc/inittab:

- - S:2345:respawn:/sbin/getty 115200 tty02
- + S:2345:respawn:/sbin/getty 115200 ttyS2



Partition	File System	Label	Size	Used	Unused	Flags
/dev/sdb1	🟩 fat16	boot	60.00 MiB	138.00 KiB	59.87 MiB	boot, lba
/dev/sdb2	🟦 ext3	rootfs	1.79 GiB	63.49 MiB	1.72 GiB	

0 operations pending

Figure D.1 – SD Card partitions

Stellenbosch University http://scholar.sun.ac.za

Bibliography

[1] Pixhawk gumstix camera makes it to product! @ONLINE. March 2011. Available at: `http://www.diydrones.com/profiles/blogs/pixhawk-gumstix-camera-makes`

[2] Ground effect @ONLINE. July 2012. Available at: `http://www.answers.com/topic/ground-effect-2`

[3] Latest aviation images @ONLINE. June 2009. Available at: `http://www.aviationspectator.com/image/latest-aviation-images`

[4] Ah-1 supercobra landing @ONLINE. August 2007. Available at: `http://www.hoveringhelicopter.com/helicopterimages/bell-military-helicopters/ah-1-supercobra-landing`

[5] Rossouw, E.: *Autonomous flight of an unmanned helicopter.* Master's thesis, Unversity of Stellenbosch, 2008.

[6] Visser, B.: *Precision landing of an UAV.* Master's thesis, Unversity of Stellenbosch, 2008.

[7] Schalkwyk, C.V.: *Full state control of a Fury X-Cell unmanned helicopter.* Master's thesis, University of Stellenbosch, 2009.

[8] Smit, P.: *Development of a 3-DOF motion simulation platform.* Master's thesis, University of Stellenbosch, 2010.

[9] de Jager, A.: *The design and implementation of vision-based autonomous rotorcraft landing.* Master's thesis, University of Stellenbosch, 2011.

[10] Garratt, D.M., Pota, A.H., Lambert, D.A., Eckersley-Maslin, L.S. and Farabet, M.C.: Visual tracking and lidar relative positioning for automated launch and recovery of an unmanned rotorcraft from ships at sea.

Stellenbosch University http://scholar.sun.ac.za

[11] Barron, J. and Thacker, N.: *Tutorial: Computing 2D and 3D Optical Flow.* Master's thesis, Medical School, University of Manchester, 2005.

[12] Zingg, S., Scaramuzza, D., Weiss, S. and Siegwart, R.: *MAV Navigation through Indoor Corridors Using Optical Flow.* Master's thesis, Autonomous Systems Lab, ETH Zurich, 2010.

[13] Zamudio, Z., Lozano, R., Torres, J. and Campos, E.: *Stabilization of a Helicopter Using Optical Flow.* Master's thesis, Automatic Control Department, CINVESTAV-IPN, México D.F., México, 2007.

[14] Stowers, J., Bainbridge-Smith, A., Hayes, M. and Mills, S.: *Optical Flow for Heading Estimation of a Quadrotor Helicopter.* Master's thesis, University of Canterbury, New Zealand, 2009.

[15] Hartley, R. and Zisserman, A.: *Multiple View Geometry in Computer Vision.* Cambridge University Press, Second Edition, 2003.

[16] Xu, C., Qiu, L., Liu, M., Kong, B. and Ge, Y.: Stereo vision based relative pose and motion estimation for unmanned helicopter landing. In: *International Conference on Information Acquisition, Proceedings of the 2006 IEEE.* 2006.

[17] Johnson, A., Montgomery, J. and Matthies, L.: *Vision Guided Landing of an Autonomous Helicopter in Hazardous Terrain.* Master's thesis, Jet Propulsion Laboratory, California Institute of Technology, 2006.

[18] Weiss, S., Scaramuzza, D. and Siegwart, R.: *Monocular-SLAMBased Navigation for Autonomous Micro Helicopters in GPS-Denied Environments.* Master's thesis, Autonomous Systems Lab, ETH Zurich, Zurich 8092, Switzerland, 2011.

[19] Hough transform @ONLINE. January 2007.
Available at: `http://www.answers.com/topic/hough-transform`

[20] Yuan, B. and Hao, Y.: A method of vision-based state estimation of an unmanned helicopter. Tech. Rep., Dalian University of Technology, 2008.

[21] Saripalli, S., Montgomery, J.F. and Sukhatme, G.S.: Vision-based autonomous landing of an unmanned aerial vehicle.

[22] Saripalli, S. and Sukhatme, G.S.: Landing on a moving target using an autonomous helicopter.

Stellenbosch University http://scholar.sun.ac.za

[23] Yang, X., Pota, H., Garratt, M. and Ugrinovskii, V.: Ship motion prediction for maritime flight operations. In: *The International Federation of Automatic Control*. 2008.

[24] Cinneide, A.O.: *Linear Prediction: The Technique, Its Solution and Application to Speech*. Master's thesis, Dublin Institute of Technology, 2008.

[25] Markel, J. and Gray, A.: *Linear Prediction of Speech*. Springer-Verlag, 1976.

[26] Garg, M.: Linear prediction algorithms. Tech. Rep., Indian Institute of Technology, Bombay,India, 2003.

[27] KOSANAM, S.: *KALMAN FILTERING FOR UNCERTAIN NOISE COVARIANCES*. Master's thesis, Andhra University, India, 2000.

[28] Khan, A., Bil, A.C. and Marion, K.E.: Real time prediction of ship motion for the aid of helicopter and aircraft deployment and recovery. In: *ICAS*. 2006.

[29] Qi, L., Qian, L., StephenWoodruff and Cartes, D.: Prony analysis for power systemtransient harmonics. *EURASIP Journal on Advances in Signal Processing*, vol. Volume 2007.

[30] Haykin, S.: *Adaptive Filter Theory, Fourth Edition*. Prentice Hall Inc, 2002.

[31] Zhao, X., Xu, R. and Kwan, C.: Ship-motion prediction: Algorithms and simulation results.

[32] Shamma, J.S.: Existence of sdre stabilizing feedback. In: *American Control Conference*. 2001.

[33] Morris, K. and Navasca, C.: Iterative solution of algebraic riccati equations for damped systems.

[34] Molenaar, P.Z.: *Model Predictive Control to Autonomous Helicopter Flight*. Master's thesis, Aalborg University, 2006.

[35] Mayne, D.Q., Rawlings, J.B., Rao, C.V. and Scokaert, P.O.M.: Constrained model predictive control: Stability and optimality. *Automatica*, vol. 36, pp. 789–814, 2000.

[36] Samal, M.K., Garratt, M., Pota, H. and Sangani, H.T.: Model predictive attitude control of vario unmanned helicopter.

[37] Saghafi, F. and Esmailifar, S.M.: Autonomatic landing of small helicopters on 4dof moving platform. *JAST*, vol. 4, 2007.

Stellenbosch University http://scholar.sun.ac.za

[38] Balderud, J.: *Modelling and Control of a Toy-Helicopter Master's Thesis*. Master's thesis, 2002.

[39] Mohammadzaheri, M. and Chen, L.: Design of an intelligent controller for a model helicopter using neuro-predictive method with fuzzy compensation. *World Congress on Engineering*, vol. 1, 2007.

[40] Wan, E.A., Bogdanov, A.A., Kieburtz, R., Baptista, A., Carlsson, M., Zhang, Y. and Zulauf, M.: Model predictive neural control for aggressive helicopter maneuvers. Tech. Rep., OGI School of Science and Engineering, 2008.

[41] Suzuki, S. and Abe, K.: Topological structural analysis of digitized binary images by border following. *COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING*, vol. 30, pp. 32–46, 1985.

[42] WU, S.T. and M´ARQUEZ, M.R.G.: *A non-self-intersection Douglas-Peucker Algorithm*. Master's thesis, School of Electrical and Computer Engineering, State University of Campinas.

[43] Mettler, B., Tischler, M.B. and Kanade, T.: System identification of small-size unmanned helicopter dynamics. *American Helicopter Society*, vol. 55, 1999.

[44] Neumann, J.B., Manz, A., Ford, T.J. and Mulyk, O.: Test results from a new 2 cm real time kinematic gps positioning system. In: *ION GPS96*. 1996.

[45] Ford, T.J. and Neumann, J.: Novatels rt-20 a real time floating ambiguity positioning system. In: *ION GPS-94*. 1996.

[46] Pulla, D.P.: *A STUDY OF HELICOPTER AERODYNAMICS IN GROUND EFFECT*. Ph.D. thesis, The Ohio State University, 2006.

[47] *MIRCOCHIP : PIC32MX3XX/4XX Family Data Sheet 64/100-Pin General Purpose and USB 32-Bit Flash Microcontrollers*.

[48] finally! - a very small linux machine @ONLINE. April 2004.
Available at: `http://web.archive.org/web/20040408235922/http://www.gumstix.org/index.html`

[49] Abount gumstix, inc @ONLINE. January 2004.
Available at: `https://www.gumstix.com/about.html`

[50] How to @ONLINE. January 2012.
Available at: `http://gumstix.org/software-development/how-to`

Stellenbosch University http://scholar.sun.ac.za

[51] Openembedded developers @ONLINE. February 2004.
Available at: `http://www.openembedded.org/index.php/OpenEmbedded_Developers`

[52] Project: Bitbake build tool - summary @ONLINE. April 2005.
Available at: `http://developer.berlios.de/projects/bitbake`

[53] Caspa camera boards @ONLINE. January 2012.
Available at: `http://wiki.gumstix.org/index.php?title=Caspa_camera_boards`

[54] Jumpnow @ONLINE. January 2012.
Available at: `http://www.jumpnowtek.com`

[55] gumsnap @ONLINE. January 2012.
Available at: `https://github.com/scottellis/gumsnap`