Fault Diagnosis and Accommodation in Quadrotor Simultaneous Localization and Mapping Systems

Anthony J. Green
Wright State University

Follow this and additional works at: https://corescholar.libraries.wright.edu/etd_all

Part of the Electrical and Computer Engineering Commons

Repository Citation
Green, Anthony J., "Fault Diagnosis and Accommodation in Quadrotor Simultaneous Localization and Mapping Systems" (2023). Browse all Theses and Dissertations. 2808.
https://corescholar.libraries.wright.edu/etd_all/2808

This Thesis is brought to you for free and open access by the Theses and Dissertations at CORE Scholar. It has been accepted for inclusion in Browse all Theses and Dissertations by an authorized administrator of CORE Scholar. For more information, please contact library-corescholar@wright.edu.
FAULT DIAGNOSIS AND ACCOMMODATION IN QUADROTOR SIMULTANEOUS LOCALIZATION AND MAPPING SYSTEMS

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Electrical Engineering

By

Anthony J. Green

B.S.E.E, Wright State University, 2021

2023

Wright State University
WRIGHT STATE UNIVERSITY
COLLEGE OF GRADUATE PROGRAMS AND HONORS STUDIES

04/28/2023


_____________________________
Xiaodong Zhang, Ph.D.
Thesis Director

_____________________________
Michael A. Saville, Ph.D., P.E.
Chair, Electrical Engineering

Committee on Final Examination:

________________________________
Zhiqiang Wu, Ph.D.

________________________________
Weisong Wang, Ph.D.

________________________________
Shu Schiller, Ph.D.
Interim Dean, College of
Graduate Programs & Honors Studies
Abstract:

Green, Anthony J. M.S.E.E. Department of Electrical Engineering, Wright State University 2023.

Fault Diagnosis and Accommodation in Quadrotor Simultaneous Localization and Mapping Systems.

Simultaneous Localization and Mapping (SLAM) is the process of using distance measurements to points in the surrounding environment to build a digital map and perform localization. It has been observed that featureless environments like tunnels or straight hallways will cause positioning faults in SLAM. This research investigates the fault diagnosis and accommodation problem for a laser-rangefinder-based SLAM systems on a quadrotor. A potential solution of using optical flow as velocity estimate and an extended Kalman filter (EKF) to perform position estimation is proposed. A fault diagnosis method for detecting faults in positional SLAM data or optical flow velocity data is developed by using two parallel EKFs. When a fault in the SLAM position or optical flow velocity is detected, the EKF adapts to provide a robust position estimate to ensure the safety of the flight control system.
# Table of Contents

1 Introduction .............................................................................................................. 1

1.1 Research Motivation .......................................................................................... 1
1.2 Research objectives ........................................................................................... 2
1.3 Thesis organization ............................................................................................ 2

2 SLAM and Optical flow ....................................................................................... 5

2.1 SLAM .................................................................................................................. 5
   2.1.1 SLAM BACKGROUND ............................................................................... 5
   2.1.2 SLAM IMPLEMENTATION ...................................................................... 6
2.2 VICON ................................................................................................................ 8
   2.2.1 VICON BACKGROUND .......................................................................... 8
   2.2.2 VICON SETUP ....................................................................................... 9
2.3 Optical Flow ....................................................................................................... 11
   2.3.1 Optical Flow Background ....................................................................... 11
   2.3.2 Optical Flow Setup ................................................................................. 15

3 EKF ....................................................................................................................... 19

3.1 EKF Background ............................................................................................... 19
3.2 Modeling ............................................................................................................ 20
3.3 EKF Setup .......................................................................................................... 22
3.4 Nominal case ...................................................................................................... 27
   3.4.1 Control EKF .......................................................................................... 29
   3.4.2 SLAM EKF ............................................................................................ 33
   3.4.3 Optical Flow EKF .................................................................................. 37

4 Fault Diagnosis Algorithm .................................................................................. 41

4.1 SLAM and Optical flow sensor faults ................................................................. 41
4.2 Sensor Fault Diagnosis Algorithm .................................................................... 42

5 Results .................................................................................................................. 47

5.1 Nominal Case ................................................................................................... 47
5.2 SLAM Freezing ................................................................................................. 51
List of Figures

Figure 1 Qdrone RQT graph showing the ROS nodes, publishers, and listeners ......................................................... 6
Figure 2 Map that can be used for both X and Y localization ................................................................................. 7
Figure 3 Map that will likely lead to a faulted X localization .................................................................................. 8
Figure 4 Drone object made from reflector points .................................................................................................. 10
Figure 5 Image showing reflector points mounted on the drone ................................................................................ 11
Figure 6 First image used in optical flow calculation .......................................................................................... 13
Figure 7 Second image used in optical flow calculation ........................................................................................ 14
Figure 8 Showing possible directions of movement between first image and second image ................................. 14
Figure 9 Plot showing a comparison of Vicon x velocity to optical flow x velocity ............................................ 17
Figure 10 Plot showing a comparison of Vicon y velocity to optical flow y velocity ............................................ 18
Figure 11 X position comparison between control EKF, command signal, and VICON ....................................... 29
Figure 12 Y position comparison between control EKF, command signal, and VICON ....................................... 30
Figure 13 Velocity comparison between control EKF and VICON ........................................................................ 31
Figure 14 Y velocity comparison between control EKF and VICON .................................................................... 32
Figure 15 X position comparison between SLAM EKF, SLAM reading, and VICON ........................................... 33
Figure 16 Y position comparison between SLAM EKF, SLAM reading, and VICON ........................................... 34
Figure 17 X velocity comparison between SLAM EKF and VICON .................................................................... 35
Figure 18 Y velocity comparison between SLAM EKF and VICON .................................................................... 36
Figure 19 X position comparison between Optical Flow EKF and VICON ............................................................. 37
Figure 20 Y position comparison between Optical Flow EKF and VICON ............................................................. 38
Figure 21 X velocity comparison between Optical Flow EKF, Optical Flow measurement, and VICON .......... 39
Figure 22 Y velocity comparison between Optical Flow EKF, Optical Flow measurement, and VICON .......... 40
Figure 23 Sensor fault diagnosis and accommodation architecture ......................................................................... 46
Figure 24 X position estimate comparison between the SLAM EKF and optical flow EKF ................................. 47
Figure 25 Y position estimate comparison between the SLAM EKF and optical flow EKF ................................. 48
Figure 26 X position estimate residual between the SLAM EKF and optical flow EKF ..................................... 49
Figure 27 Y position estimate residual between the SLAM EKF and optical flow EKF ..................................... 49
Figure 28 Graph comparing the x position estimates and measurements during a frozen SLAM x position condition ........................................................................................................................................... 51
Figure 29 Zoomed in graph comparing the x position estimates and measurements during a frozen SLAM x position condition ........................................................................................................................................... 52
Figure 30 Graph comparing the x position estimates and measurements during a frozen SLAM x position condition to actual drone position ......................................................................................... 53
Figure 31 Graph comparing the y position estimates and measurements during a frozen SLAM x position condition to actual drone position ......................................................................................... 54
Figure 32 Graph comparing the x velocity estimates and measurements during a frozen SLAM x position condition to actual drone velocity ......................................................................................... 55
Figure 33 Zoomed in graph comparing the x velocity estimates and measurements during a frozen SLAM x position condition to actual drone velocity ......................................................................................... 56
Figure 34 Graph comparing the y velocity estimates and measurements during a frozen SLAM x position condition to actual drone velocity ......................................................................................... 57
Figure 35 graph showing the x position comparison between the position estimates and measurements with a SLAM bias ................................................................. 60
Figure 36 graph showing the absolute value of the filtered difference in x position ................................................. 61
Figure 37 zoomed in graph showing the x position comparison between the position estimates and measurements with a SLAM bias ................................................................. 62
Figure 38 graph showing the y position comparison between the position estimates and measurements with a SLAM bias ........................................................................... 63
Figure 39 graph showing the absolute value of the filtered difference in y position .................................................. 64
Figure 40 zoomed in graph showing the y position comparison between the position estimates and measurements with a SLAM bias ........................................................................... 65
Figure 41 graph showing the x velocity comparison between the velocity estimates and measurements with a SLAM bias ............................................................................................ 66
Figure 42 graph showing the absolute value of the filtered difference in x velocity .................................................. 67
Figure 43 graph showing the y velocity comparison between the velocity estimates and measurements with a SLAM bias ............................................................................................ 68
Figure 44 graph showing the absolute value of the filtered difference in y velocity .................................................. 69
Figure 45 graph showing the x position comparison between the position estimates and measurements with an optical flow bias ........................................................................... 70
Figure 46 graph showing the absolute value of the filtered difference in the x position ........................................... 71
Figure 47 zoomed in graph showing the x position comparison between the position estimates and measurements with an optical flow bias ........................................................................... 72
Figure 48 graph showing the y position comparison between the position estimates and measurements with an optical flow bias ........................................................................... 73
Figure 49 graph showing the absolute value of the filtered difference in the y position ........................................... 74
Figure 50 zoomed in graph showing the y position comparison between the position estimates and measurements with an optical flow bias ........................................................................... 75
Figure 51 graph showing the x velocity comparison between the velocity estimates and measurements with an optical flow bias ............................................................................................ 76
Figure 52 graph showing the absolute value of the filtered difference in the x velocity ........................................... 77
Figure 53 graph showing the y velocity comparison between the velocity estimates and measurements with an optical flow bias ............................................................................................ 78
Figure 54 graph showing the absolute value of the filtered difference in the y velocity ........................................... 79
Figure 55 graph showing the x position comparison between the position estimates and measurements with a frozen optical flow reading ........................................................................... 80
Figure 56 graph showing the absolute value of the filtered difference in the x position ........................................... 81
Figure 57 zoomed in graph showing the x position comparison between the position estimates and measurements with a frozen optical flow reading ........................................................................... 82
Figure 58 graph showing the y position comparison between the position estimates and measurements with a frozen optical flow reading ........................................................................... 83
Figure 59 graph showing the absolute value of the filtered difference in the y position ........................................... 84
Figure 60 graph showing the y position comparison between the position estimates and measurements with a frozen optical flow reading ........................................................................... 85
Figure 61 zoomed in graph showing the x velocity comparison between the position estimates and measurements with a frozen optical flow reading ........................................................................... 86
Figure 62 graph showing the absolute filtered difference in the x velocity ................................................................. 87

vii
Figure 63 graph showing the y velocity comparison between the position estimates and measurements with a frozen optical flow reading ................................................................................................................................. 92
Figure 64 graph showing the absolute value of the filtered difference in the y velocity ................................................................. 93
Acknowledgements:

I would like to take a moment to thank everyone that has been instrumental in helping me do this research and complete my masters degree. I would like to thank Professor Zhang Xiaodong for putting up with me in the six classes of his that I have taken, the undergraduate research summer program, and being my thesis advisor. He is always very patient and willing to reexplain the same concept multiple times over until I have a grasp of it. I could not have done this without him pointing me in the right direction so many times.

I want to thank Jason Gauthier for teaching me how to use the drone, SLAM, VICON, and the Simulink commander. I also want to thank him for coming in over the weekend when I was having problems with the drone to troubleshoot with me.

I want to thank my parents Malissa and Francisco Green for providing a warm and loving home for me during all these years of college. There is no way that I could have continued to graduate school if it had not been for their overwhelming support.

I want to thank my work supervisors Brad Miller and Deb Herman for allowing me to adjust my work schedule whenever I needed extra time for school.

Lastly, I want to thank God for showering me with uncountable blessings and providing me the opportunities that I have had so far.
1 Introduction

1.1 Research Motivation

In recent decades, Unmanned Aerial Vehicles (UAVs) have attracted significant attentions due to their potentials in various military and civilian applications [1-3]. Simultaneous Localization and Mapping (SLAM) is an important system for unmanned autonomous systems. While determining position is possible from methods such as GPS and other methods of external tracking or references, these methods have one common problem in that they are external to the autonomous system.

To design a system that can operate in many different environments the system cannot be tied to setting up the environment first. This is where SLAM becomes useful. Rather than receiving time data from different sources and triangulating its position (GPS), or being sent its position by some external source, SLAM uses sensors attached to the autonomous system to determine how far it is from points in its environment, determine its localized position, and build a map of its surroundings.

Research activities have shown very promising results for autonomous quadrotor mapping and navigation using SLAM [4-6]. This works well under many conditions, however there are cases where SLAM can fail to determine its position. For example, it has been observed that featureless environments like tunnels or straight hallways will cause positioning faults in SLAM [1]. Under these circumstances it is important to have the ability to detect faults in the
SLAM system and to have a backup system to fall back on to, at the very least, keep the control system stable.

1.2 Research objectives

The primary objective of this thesis is to explore the possibility of integrating optical flow-based velocity estimates into an existing Simultaneous Localization and Mapping system to improve stability and fault tolerance during situations where the SLAM measurement is faulted due to environmental ambiguity or sensor faults. The specific research objectives include the following:

- Develop an extended Kalman filter based algorithm for providing accurate quadrotor position estimation using both SLAM position and optical flow velocity measurements during normal flight.
- Develop a fault diagnosis algorithm for detecting and isolating faults in the SLAM position measurement or optical flow velocity measurement.
- Develop a fault accommodation method to provide robust UAV position estimate after detecting a fault in the SLAM or optical flow measurement, ensuring the safety of the flight control system.
- Implement the proposed algorithms and evaluate the effectiveness using real-time flight tests.

1.3 Thesis organization
• Chapter 1: This chapter explains the problem along with the research objectives covered in this thesis.

• Chapter 2: This chapter explains the systems that are used for velocity and position measurements. It briefly covers the background on how they work and then details how they are setup for this research.

• Chapter 3: This chapter details the background and explanation of the first research objective mainly the development of an EKF to provide accurate position and velocity for flight control.

• Chapter 4: This chapter details the explanation and development of the fault detection algorithm and adaptation to the control system to enable stable flight during fault conditions.

• Chapter 5: This chapter details test flights for the nominal flight, SLAM frozen, SLAM biased, Optical Flow frozen, and Optical Flow biased cases. It does analysis of the results and summarizes the results for each case.

• Chapter 6: This chapter summarizes what was covered in this thesis and whether the research objectives were met.

• Chapter 7: This chapter details some additional research that could be done to branch off of the solutions developed here or to further improve the usability of SLAM on the drone in ambiguous environments.
• Chapter 8: This lists the references used when designing the control and fault detection algorithms and when writing this thesis.
2 SLAM and Optical flow

2.1 SLAM

2.1.1 SLAM BACKGROUND

Simultaneous Localization and Mapping, abbreviated SLAM, is the process of building a digital map and finding position using angle and distance measurements often combined with accelerometer, gyroscope, and odometer readings. This can be done both in two dimensions and three dimensions. This thesis exclusively utilizes the two-dimensional case.

For the two-dimensional case, a LIDAR system, or some other rotating distance sensor, is used to measure the distance at many different angles making a point cloud. This point cloud is then sent to a scan matcher and a mapping algorithm often along with some other odometry or accelerometer data. The mapping algorithm and scan matcher work together to match the data to any preexisting map and further build on a map of the surroundings. The scan matcher additionally uses the point cloud and the map to determine where in the map the measurement location came from, thereby determining the sensor location.

There are several different mapping algorithms that can be used to build the map and determine position. Some of these algorithms include Hector SLAM [7], Karto [8], and Gmapping [9]. For this research a SLAM system using Gmapping was used for the map building and positioning.

Gmapping is a technique that uses Roa-Blackewllized Particle Filters to reduce a distribution of particles each containing a map of the environment [10]. The general flow of the algorithm is as follows [11, 12]:

5
1) A particle filter is used to estimate the pose of the robot and the map of the environment. The filter represents the posterior distribution of the pose and map.

2) The algorithm uses a sensor model to estimate the probability of sensor measurements given the current particles.

3) Update the map of the environment using the measurements.

4) Select the most accurate particles to be used in the next iteration and discard the other particles.

5) Update the distribution of the robot state and map.

2.1.2 SLAM IMPLEMENTATION

ROS, short for Robot Operating System, is used on the Qdrone to produce a SLAM map. The drone sends IMU data (accelerometer and gyroscope) to the pose2d server which is published to the IMU node. The laser scan matcher then uses the IMU information combined with the LIDAR scan data and the current map transform to determine the current location. This location information is sent to the map transform and to the pose2d server, so that the drone can update its position. Gmapping then uses the current location data along with the LIDAR scan to update the current map.

Figure 1 Qdrone RQT graph showing the ROS nodes, publishers, and listeners
Figure 1 shows the RQT graph of the nodes, publishers, and listeners that run on the Qdrone during a flight. The urg_done node is the laser scan data coming from the LIDAR (light imaging, detection, and ranging) while the pose2d_server node is used to communicate the 2d pose information to the drone and communicate the IMU data to the laser scan matcher.

Figure 2 Map that can be used for both X and Y localization

Figure 2 shows a representation of a usable map built by SLAM using LIDAR data. Note that there are objects that can be used for localization in both the Y direction (up and down on the map) and the X direction (right and left on the map).
It has been observed that featureless environments like tunnels or straight hallways will cause positioning faults in SLAM [1]. Figure 3 shows an example of a map that can cause a localization fault. The drone will still be able to localize in the Y position. However, there are no objects that it can accurately determine the X position from. This will potentially cause the X signal to go “stale” causing uncontrollability and instability of the flight control system in the X direction.

2.2 VICON

2.2.1 VICON BACKGROUND

VICON is a brand of motion tracking system that uses multiple high resolution infrared cameras to track infrared reflective objects in a predefined area. The cameras are calibrated with a light stick that has a precise known distance between each of the calibration points. Using this stick, the origin is set and a map of points and distances is made for the cameras. The cameras are then able to triangulate an exact three-dimensional point relative to the origin provided that at least three cameras can see any infrared tracking reflector at a given point in time. VICON
provides a software that can be used to make a model of reflector points and then, provided that enough of the points are visible at any given time, can determine the position of the model in three-dimensional space.

By tracking how the points move over time it can also determine the velocity and acceleration of the points. Based on how the points are oriented relative to the origin, the VICON system can determine the model’s roll, pitch, yaw, and angular velocities.

2.2.2 VICON SETUP

In the drone lab there are a total of four infrared tracking cameras placed at the corners of the flight cage. These cameras are connected to a control box that is connected to a desktop
which runs the VICON software. A model of the drone, see Figure 4, is made based on the location of the infrared reflectors which can be seen in Figure 5.

*Figure 4 drone object made from reflector points*
Figure 5 image showing reflector points mounted on the drone

The vehicle’s position and angles are then sent to a VICON stream client that is hosted on an ubuntu virtual machine. The data is then combined with the remote-control instructions for the drone and sent to the Simulink commander running on the drone.

2.3 Optical Flow

2.3.1 Optical Flow Background

Optical Flow, abbreviated OF, is the process of determining movement based on the average pixel movement between consecutive frames in a video stream. A brief description of the velocity estimation algorithm using optical flow is given below [13-15]. If the brightness of the image is relatively constant over time, then a pixel at a location of \((x, y, t)\) will have an intensity...
of I(x, y, t). When moved by an amount of Δx, Δy, and Δt between consecutive camera frames will result in the intensity of:

\[ I(x + Δx, y + Δy, t + Δt) ≈ f(x, y, t) \]

If the movement between frames is small then this can be rewritten using Taylor series to get approximately:

\[ \frac{∂I}{∂x} Δx + \frac{∂I}{∂y} Δy + \frac{∂I}{∂t} Δt = 0 \]

which can be divided by Δt to get:

\[ \frac{∂I}{∂x} \frac{Δx}{Δt} + \frac{∂I}{∂y} \frac{Δy}{Δt} + \frac{∂I}{∂t} = 0 \]

This is the same as the equation:

\[ \frac{∂I}{∂x} V_x + \frac{∂I}{∂y} V_y + \frac{∂I}{∂t} = 0 \]

where \( V_x \) and \( V_y \) are the velocity vectors at the x and y positions respectively, and \( \frac{∂I}{∂x}, \frac{∂I}{∂y}, \frac{∂I}{∂t} \) and \( \frac{∂I}{∂y} \) are the partial derivatives of the image function in their corresponding derivatives. Using this information one final equation can be written:

\[ ΔI \ast v + It = 0 \text{ or } ΔI \ast v = −It \]

with \( v \) being the velocity vector, \( It \) being the time, and \( ΔI \) being the spatial gradient. The time \( It \) is the time between camera frames, and the spatial gradient can be calculated using partial derivatives, leaving just the velocity vector which is the desired unknown. This means that if there is a constant brightness, any change in pixel gradient will be caused by some pixel velocity or flow between camera frames. The problem now is that there is one equation with two different
unknowns which means that, without additional constraining equations, the velocity vector is not uniquely determinable for one pixel.

This problem is known as the aperture problem and states that unless you can see the entire shape or outline of a particular object it is impossible to know for certain whether any change in the shape is due to a movement in the x direction, y direction, or some combination of the two. To illustrate this point, look at the picture shown in Figure 6.

![Figure 6 first image used in optical flow calculation](image)

Imagine that the black circle is a hole and the red line is an object that is being tracked through consecutive camera frames. Suppose that the next camera frame is the picture in Figure 7.
Figure 7 second image used in optical flow calculation

From comparing this frame to the last frame, which direction did the red line move? Without knowing more information it is impossible to tell. It could be either of the following or some combination of the two (see Figure 8).

Figure 8 showing possible directions of movement between first image and second image

There are several ways to get around this under-constrained problem. One of the ways is to only look at corners of objects where both x and y movement can be uniquely determined. This method is known as feature finding [13] and can be useful if the camera is pointing at a
feature rich environment. This method falls short when the environment lacks distinguishing features like, for example, a uniform black foam mat to provide a soft landing for a drone.

Another method of solving the constraints problem is by using more than just one pixel to solve for the velocity vector. Rather than considering one pixel, the least squared errors approach is used to find the velocity vector that minimizes the error of a block of pixels together [13]. Repeating this procedure for blocks of pixels over the entire image results in the average vector velocity of the image.

A third way of computing the optical flow [16] is to take a group of pixels, also known as a tile, and then compare its gradient to the surrounding areas in the next frames image. The optical flow algorithm computes the sum of absolute differences with each possible pair and then if the sum of absolute differences is less than a value threshold it considers it a match. This is then done for a number of different tiles in each frame and the pixel velocity is calculated. This becomes very computationally expensive if the search area around the original tile is large.

For any of these methods if the distance to the surface or object in the image is known, and the velocity of the surface or object is known, then the velocity of the camera can be found. There are ways to determine this relationship given the aperture and focal point of the camera. However, it can also be experimentally determined if the aperture and focal point information is unknown.

2.3.2 Optical Flow Setup

The Qdrone quadrotor system used for experimentation in this thesis has a OmniVision OV7251 downward facing camera built into the platform that can run at up to 640*480 pixels
and up to 120 FPS. For optical flow, the processing power and processing time required to compute the flow increases quickly as the resolution of the image increases.

For this research, the optical flow camera was set at a capture resolution of 240*240 pixels at 30 frames per second. The video stream from the camera is then fed into a Quanser program optical flow Simulink block which then determines the optical flow pixel velocities of the camera stream. The optical flow block used the global algorithm [17] which is based on the sum of absolute differences approach, the third method described, with 10000 tiles sized at 8*8 pixels with a feature threshold of 100 and a value threshold of 3000. The search size used was 10 pixels.

The feature threshold is used to determine if a particular area is usable for optical flow calculation. If the average pixel gradient is less than this value, then the tile is discarded. The value threshold is used to determine if a tile matches with a tile from the previous image. The sum of absolute differences must be less than the threshold. The search size is how far the optical flow algorithm will look around the current tile for matches. The processing speed of the algorithm drops off quickly as this parameter increases.

The tile quantity being 10000 per image was not a computational problem for the drone because of the thresholds that quickly narrow down which pixel groups are suitable to be used for the search. Initially some tests were done with higher resolution images. However, those resulted in more sensitivity to small movements that lost the ability to track larger movements and velocities.
The relationship of pixel velocities to drone velocity was experimentally found by using the VICON tracking system to compare the measured drone velocity to the optical flow velocity. This relationship was approximated as:

\[
\text{Optical Flow} \times 0.10 \times \text{drone height} = \text{drone velocity}
\]

when using a camera resolution of 240*240 and the previously mentioned optical flow parameters. The optical flow was measured in pixels per second, and the drone velocity was in meters per second.

*Figure 9 plot showing a comparison of Vicon x velocity to optical flow x velocity*
Figures 9 and 10 shows the comparison between the optical flow velocity estimate and the VICON velocity measurement which is considered to be the ground truth. As can be seen, the optical flow velocity approximates the ground truth VICON velocities well. Note that the optical flow is forced to be zero whenever the drone is not in the “flight mode” state. This is done to prevent unpredictable velocity readings when the drone is landing or tacking off due to the expanding or contracting motion of the pixels. The optical flow reading will also freeze or hold the last value if the camera stream lags for a frame to prevent sudden jumps to zero. The optical flow signal is passed through a low pass filter to reduce noise introduced by film grain and vibrations. There is still a significant noise level in the resulting velocity estimate, however reducing the cutoff filter beyond a certain point causes significant problems with the derivative position control due to the increased time delay.
3 EKF

3.1 EKF Background

The drone has several sensors that each update at different rates. To deal with the potential problem of information being available at different times, and to fuse the complementary information provided by different sensors, the drone uses an Extended Kalman Filter (EKF) to combine all the sensor data and track the state of the drone.

An EKF can conceptually be split into two different processes. The first process is the prediction step where a system model is used to predict the states at the next time step given the current timesteps values. The following equations describe the prediction step:

\[
\hat{x}^{-} = f(\hat{x}, u) \\
P^{-} = AP + PA^T + Q \\
A = \frac{\partial f}{\partial x}(\hat{x}, u)
\]

where \( f(\hat{x}, u) \) is the system model which is typically nonlinear, \( A \) is the Jacobian matrix, the partial derivative of the function \( f(\hat{x}, u) \) with respect to each state, \( P \) is an error covariance matrix, and \( Q \) is a noise covariance matrix for each of the states. \( P \) is updated during the second part of the process.

The second part of the process is the update step where the states are updated based on the sensor readings. This can happen periodically or whenever there is a new sensor reading. If the sensors have different refresh rates, then this can also happen at different intervals for different sensors. The update process uses the following equations:
\[ K = P - C^T (R + C P - C^T)^{-1} \]

\[ \hat{x} = \hat{x}^- + K (y - C(\hat{x}^-)) \]

\[ P = (I - KC)P^- \]

where \( R \) is a noise covariance matrix that is set based on the sensor precision and accuracy, \( P \) is from the prediction part of the process, \( C \) is a Jacobian matrix of the partial derivatives of each state that is being updated, and \( K \) is the Kalman gain that controls how much the states that are being updated change by, \( \hat{x}^- \) is \( \hat{x} \) from the previous iteration with \( \hat{x} \) being all the state estimates, and \( y \) is the update vector containing the sensor data that is being used to update the model.

### 3.2 Modeling

As explained in [18-19], the quadrotor state vector includes twelve state variables:

\[ x = [p_x, p_y, p_z, u, v, w, \phi, \theta, \psi, p, q, r]^T \]

where \( p_x, p_y, p_z \) represent the position of the quadrotor in the inertial frame, \( u, v, w \), represent the body velocity of the quadrotor, the angles \( \phi, \theta, \) and \( \psi \), are the roll, pitch and yaw angles about the \( x, y \) and \( z \) axis, respectively, and \( p, q, r \), are the body angular rates. Based on the Newton-Euler equations of motion, the quadrotor state space model is represented by the following equations: [18]:

\[
\begin{bmatrix}
\dot{p}_x \\
\dot{p}_y \\
\dot{p}_z
\end{bmatrix} = R^e_{b}(\psi, \theta, \phi) 
\begin{bmatrix}
u \\
v \\
w
\end{bmatrix}
\]
\[
\begin{bmatrix}
\dot{u} \\
\dot{v} \\
\dot{w}
\end{bmatrix} =
\begin{bmatrix}
rv - qw \\
pw - ru \\
qu - pv
\end{bmatrix} + \begin{bmatrix}
-gsin(\theta) \\
gcos(\theta)sin(\phi) \\
gcos(\theta)cos(\phi)
\end{bmatrix} + \frac{1}{m}\begin{bmatrix}
0 \\
0 \\
-F
\end{bmatrix}
\]

\[
\begin{bmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{bmatrix} = \begin{bmatrix}
\left(\frac{I_y - I_z}{I_x}\right)p + \frac{1}{I_x}\tau_{\phi} \\
\left(\frac{I_z - I_x}{I_y}\right)p + \frac{1}{I_y}\tau_{\theta} \\
\left(\frac{I_x - I_y}{I_z}\right)p + \frac{1}{I_z}\tau_{\psi}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} = R_{\rho}(\phi, \theta)\begin{bmatrix}p \\ q \\ r \end{bmatrix}
\]

where \(m\) is the mass of the quadrotor, \(g\) is acceleration from gravity, \(I_x, I_y, I_z\), are the moments of inertia around body axes \(x, y, z\), respectively. The inputs to the system include the thrust force \(F\) and the roll, pitch, and yaw torque represented by \(\tau_{\phi}, \tau_{\theta}, \tau_{\psi}\), respectively. Additionally, \(R_{b}\) represents the rotation matrix from the body frame to the inertial frame [18], and the matrix \(R_{\rho}\) represents the transformation between angular rates and the Euler angle rates, which are given by [19]:

\[
R_{\rho}(\phi, \theta) = \begin{bmatrix}
1 & \sin(\phi) \tan(\theta) & \cos(\phi) \tan(\theta) \\
0 & \cos(\phi) & -\sin(\phi) \\
0 & \sin(\phi) \sec(\theta) & \cos(\phi) \sec(\theta)
\end{bmatrix}
\]

\[
R_{b}^e = \begin{bmatrix}
\cos(\psi) \cos(\theta) & \cos(\psi) \sin(\phi) \sin(\theta) - \cos(\phi) \sin(\psi) & \cos(\phi) \sin(\theta) \cos(\psi) \\
\sin(\psi) \cos(\theta) & \sin(\psi) \sin(\phi) \sin(\theta) + \cos(\phi) \cos(\psi) & \cos(\phi) \sin(\psi) \sin(\theta) - \cos(\psi) \sin(\phi) \\
\sin(\theta) & \cos(\theta) \sin(\phi) & \cos(\phi) \cos(\theta)
\end{bmatrix}
\]
3.3 EKF Setup

The extended Kalman filter uses nine of the original states of the base model with an addition of six extra states for estimating the biases in the gyroscope and accelerometer measurements [20]. As a result, the total states can be represented as:

\[ x = [p_x, p_y, p_z, u, v, w, \varphi, \theta, \psi, \beta_{ib}, \beta_{ib}, \beta_{kb}, \alpha_{ib}, \alpha_{jb}, \alpha_{kb}]^T \]

Where \( p_x, p_y, p_z \) are the positions of x, y, and z in the inertial frame, \( u, v, w \) are the x, y, and z velocities in the body frame, \( \varphi, \theta, \psi \) are the roll pitch and yaw of the drone about the x, y, and z axis, \( \beta_{ib}, \beta_{ib}, \beta_{kb}, \alpha_{ib}, \alpha_{jb}, \alpha_{kb} \) are the biases of the gyroscope and the accelerometer in the body frame.

The signals available for use in the EKF are the x and y position reading from VICON or SLAM and the height sensor, the x and y velocity estimation from optical flow, the gyroscope rates, and the accelerometer readings, and the attitude angles provided by IMU estimates or Vicon. Both the accelerometer and the gyroscope measurements refresh quickly so they can be used every iteration. The position reading from VICON happens relatively quickly so it could be used every iteration. However, if SLAM is being used then the values should be checked to make sure that they are new before they are used to update the EKF. The same applies to the optical flow reading because, even though it updates much faster than SLAM, it still updates at a slower rate than the model is run at.

The sensor update vector \( y \) is defined as the following when all the measurements are present:

\[ y = [p_x, p_y, p_z, OF_x, OF_y, \psi, \alpha_{icb}, \alpha_{jcb}] \]
where $p_x, p_y$ are the X and Y position measurements from either SLAM or VICON, $p_z$ is the Z position either measured from VICON or from the distance sensor array on the drone, $OF_x, OF_y$ are the optical flow velocity estimates in the X and Y directions as estimated by the downward facing camera, $\psi$ is the yaw angle as measured from VICON or SLAM, and finally, $\alpha c_{ib}, \alpha c_{jb}$ is the acceleration measurement from the accelerometer in the body frame x and y axes.

The corresponding C matrix for this full sensor update vector is as follows:

$$C = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}$$

This matrix is then adjusted for cases when certain sensor data is not available. For example, if there is no updated position data from SLAM then the C matrix loses the first two rows and the update vector loses its first two entries resulting in the following:

$$y = \left[p_z, OF_x, OF_y, \psi, \alpha c_{ib}, \alpha c_{jb}\right]$$

$$C = \begin{bmatrix}
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}$$

This is then repeated for other cases of missing sensor data. To be specific the following cases each get their own C matrix as described below:

- Full sensor update vector
\[ y = [p_x, p_y, p_z, O_x, O_y, \psi, \alpha c_{ib}, \alpha c_{jb}] \]

- Faulted or missing y position
  \[ y = [p_x, p_z, O_x, O_y, \psi, \alpha c_{ib}, \alpha c_{jb}] \]

- Faulted or missing x position
  \[ y = [p_y, p_z, O_x, O_y, \psi, \alpha c_{ib}, \alpha c_{jb}] \]

- Faulted or missing optical flow
  \[ y = [p_x, p_y, p_z, \psi, \alpha c_{ib}, \alpha c_{jb}] \]

- No (x, y) position or velocity
  \[ y = [p_z, \psi, \alpha c_{ib}, \alpha c_{jb}] \]

The height, Yaw, and acceleration readings are always assumed to be present.

The A matrix is a 15*15 matrix which is too large to list all at once so it is listed below in sections with the column numbers being specified:

\[
A_{1 \to 5} = \begin{bmatrix}
0 & 0 & 0 & c(\theta) * c(\psi) & s(\varphi) * s(\theta) * c(\psi) - c(\varphi) * s(\psi)
0 & 0 & 0 & c(\theta) * s(\psi) & s(\varphi) * s(\theta) * s(\psi) + c(\varphi) * c(\psi)
0 & 0 & 0 & -s(\theta) & s(\varphi) * c(\theta)
0 & 0 & 0 & 0 & r
0 & 0 & 0 & -r & 0
0 & 0 & 0 & q & -p
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]
$$A_6 = \begin{bmatrix}
c(\phi) \cdot s(\theta) \cdot c(\psi) + s(\phi) \cdot s(\psi) \\
c(\phi) \cdot s(\theta) \cdot s(\psi) - s(\phi) \cdot c(\psi) \\
c(\phi) \cdot c(\theta) \\
- q \\
p \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}$$

$$A_7 = \begin{bmatrix}
(c(\phi) \cdot s(\theta) \cdot c(\psi) + s(\phi) \cdot s(\psi)) \cdot v + (-s(\phi) \cdot s(\theta) \cdot c(\psi) + c(\phi) \cdot s(\psi)) \cdot w \\
(c(\phi) \cdot s(\theta) \cdot s(\psi) - s(\phi) \cdot c(\psi)) \cdot v + (-s(\phi) \cdot s(\theta) \cdot s(\psi) - c(\phi) \cdot c(\psi)) \cdot w \\
c(\phi) \cdot c(\theta) \cdot v - s(\phi) \cdot c(\theta) \cdot w \\
0 \\
g \cdot c(\theta) \cdot c(\phi) \\
-g \cdot c(\theta) \cdot s(\phi) \\
(c(\phi) \cdot t(\theta) \cdot q) - (s(\phi) \cdot t(\theta) \cdot r) \\
(-s(\phi) \cdot q) - (c(\phi) \cdot r) \\
(c(\phi) \cdot sc(\theta) \cdot q) - (s(\phi) \cdot sc(\theta) \cdot r) \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}$$
\[
A_8 = 
\begin{vmatrix}
-s(\theta) \cdot c(\psi) \cdot u + s(\varphi) \cdot c(\theta) \cdot c(\psi) \cdot v + c(\varphi) \cdot c(\theta) \cdot c(\psi) \cdot w \\
-s(\theta) \cdot s(\psi) \cdot u + s(\varphi) \cdot c(\theta) \cdot s(\psi) \cdot v + c(\varphi) \cdot c(\theta) \cdot s(\psi) \cdot w \\
-c(\theta) \cdot u - s(\varphi) \cdot s(\theta) \cdot v - c(\varphi) \cdot s(\theta) \cdot w \\
-g \cdot c(\theta) \\
-g \cdot s(\theta) \cdot s(\varphi) \\
-g \cdot s(\theta) \cdot c(\varphi) \\
(s(\varphi) \cdot sc(\theta) \cdot q) + (c(\varphi) \cdot sc(\theta) \cdot r) \\
0 \\
(s(\varphi) \cdot sc(\theta) \cdot t(\theta) \cdot q) + (c(\varphi) \cdot sc(\theta) \cdot t(\theta) \cdot r) \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 
\end{vmatrix}
\]

\[
A_9 = 
\begin{vmatrix}
-c(\theta) \cdot s(\psi) \cdot u + (-s(\varphi) \cdot s(\theta) \cdot s(\psi) - c(\varphi) \cdot c(\psi)) \cdot v + (-c(\varphi) \cdot s(\theta) \cdot s(\psi) + s(\varphi) \cdot c(\psi)) \cdot w \\
c(\theta) \cdot c(\psi) \cdot u + (s(\varphi) \cdot s(\theta) \cdot c(\psi) - c(\varphi) \cdot c(\psi)) \cdot v + (c(\varphi) \cdot s(\theta) \cdot c(\psi) + s(\varphi) \cdot s(\psi)) \cdot w \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 
\end{vmatrix}
\]
Where $s(x)$, $c(x)$, $t(x)$, and $sc(x)$ are shorthand for $\sin(x)$, $\cos(x)$, $\tan(x)$ and $\sec(x)$ respectively.

### 3.4 Nominal case

Shown below are plots from a nominal test flight. The purpose of these plots is to show the tracking performance of the relevant states that are needed for control and fault detection compared with the VICON ground truth readings. The flight was conducted with no artificial faults inserted to provide the best-case conditions to demonstrate the capabilities of the EKFs.
The graphs are split up into three different sections. The first section is a control EKF that has access to all the sensor information. The second section is the SLAM EKF that has access to all the sensor information except the optical flow velocity reading. The final section is the optical flow EKF that has access to all the information except the SLAM position measurement. The purpose of having these three different EKFs will be discussed in Chapter 4.
3.4.1 Control EKF

*Figure 11 x position comparison between control EKF, command signal, and VICON*
Figures 11 and 12 show a comparison of the main EKF x position estimate compared with the VICON positions. The graph shows that the control EKF provides a reasonable estimate of the position for both x and y that tracks the commanded position.
Figure 13 velocity comparison between control EKF and VICON.
Figure 14 y velocity comparison between control EKF and VICON
Figures 13 and 14 show a comparison of the main EKF velocity estimates compared with the VICON velocities. This graph shows that the velocity estimates are a good approximation of the actual drone velocity.

3.4.2 SLAM EKF

*Figure 15* x position comparison between SLAM EKF, SLAM reading, and VICON
Figures 15 and 16 show the tracking performance of the position estimate coming from the SLAM EKF compared with the VICON position and the SLAM position reading. While
there are some differences, particularly for the y position SLAM EKF estimate from 37 to 42 seconds, overall, the SLAM EKF provides a good estimate of the actual drone position.

Figure 17 x velocity comparison between SLAM EKF and VICON
Figures 17 and 18 show the tracking performance of the velocity estimate coming from the SLAM EKF compared with the VICON velocities. While the x velocity estimate tracks the
ground truth well, the y velocity estimate is significantly worse. In general, the SLAM EKF provides an acceptable velocity estimate compared with the VICON ground truth.

3.4.3 Optical Flow EKF

![Figure 19 x position comparison between Optical Flow EKF and VICON](image)
Figures 19 and 20 show the tracking performance of the x and y position coming from the optical flow EKF compared with the VICON positions. While there are some differences,
notably the constant separation in the y estimate and position starting at 18 seconds, the optical flow position estimate performs well at tracking the actual drone position.

Figure 21 x velocity comparison between Optical Flow EKF, Optical Flow measurement, and VICON
Figures 21 and 22 show the tracking performance of the x and y velocity estimates coming from the optical flow EKF compared with the VICON velocities and the initial optical flow readings. The tracking performance is very good.

Based on the above graphs, all the EKFs provide good tracking performance in their position estimates. The optical flow EKF and the control EKF provide great tracking...
performance in their velocity estimates and the SLAM EKF provides usable tracking performance in its velocity estimates.

4 Fault Diagnosis Algorithm

4.1 SLAM and Optical flow sensor faults

There are several different scenarios that can cause a SLAM fault. The most obvious of these is if the laser match scanner, gmapping, or pose 2d server processes crash in ROS. This could happen for several reasons, the most common of them being the connection to the LIDAR mounted on the drone having a bad connection because the USB cable vibrated loose. When this happens the driver for the LIDAR unit cannot communicate with the hardware and crashes, causing the laser match scanner and gmapping to crash. This causes the pose2d to report the last known positions meaning that the x, y, and yaw reading coming from ROS will freeze. Another scenario is when the drone is flying in a long hallway and the LIDAR cannot see the end of the hallway. If there is not enough detail in the hallway to base an x position off, this could cause the x position value to completely freeze or slowly change at a rate not representative of the drones movements.

Optical flow methods can fail if a sufficient amount of ground-truth data is not available or when the background environment lacks distinctive features. When the optical flow algorithm stops producing an output or there is a lag in the video frames, the velocity estimation is held at its previous value or becomes completely frozen. This introduces a critical problem because the drone will think that it is moving at some constant velocity, which will quickly cause the drone to become unstable and crash.
In addition to the fault of frozen sensor measurements described above, this research also considers the occurrence of a bias fault in the SLAM position or optical flow velocity measurement. For instance, the optical flow camera could be pointing at some surface, and an error in distance estimation/measurement between the camera and the surface could cause the camera to continuously think that it is moving much faster or slower than it actually is.

4.2 Sensor Fault Diagnosis Algorithm

The drone runs three different extended Kalman filters in parallel. The main or primary EKF is used for position control and can receive all the sensor measurements. It determines what sensor measurements it uses for state estimation based on if a new sensor update is available and if there is any faulty measurement detected. The sensor fault diagnosis component consists of two extended Kalman filters running in parallel, which are used to detect and isolate faults in either the SLAM position or optical flow velocity measurement, as shown in Error! Reference source not found..

One of these two fault diagnostic EKFs estimates the quadrotor states based on the SLAM position, yaw, and accelerometer data, excluding the optical flow velocity signal, while the other fault diagnostic EKF estimate the quadrotor states based on the optical flow velocity signal, yaw, and accelerometer data, excluding the SLAM position measurement. Faults in the SLAM position measurement or optical flow signals can be detected and isolated by using the SLAM signal update status and the state estimates generated by the two diagnostic EKFs. The algorithm first checks the current update status of SLAM signal. If a new SLAM sensor update is not
available, a counter is used to determine if the SLAM signal is frozen. Otherwise, it proceeds to
the following specific procedure:

1) **Fault detection.** Under normal operating conditions, the x and y position estimates
   generated by the two diagnostic EKFs are expected to be close to each other. If the
deviation between the x or y position estimates exceeds a predefined threshold that is
selected using extensive flight data, then a sensor fault related to the SLAM signal or the
optical flow signal is detected.

2) **Fault isolation.** After fault detection, the sensor fault can be isolated by comparing the
   position and velocity estimates generated by the two diagnostic EKFs. Specifically, two
diagnostic residuals are generated, which are defined as the deviations between the
position estimates and velocity estimates provided by the two diagnostic EKFs,
respectively. Accordingly, the following fault isolation decision logic is applied:
   - If the position-based residual exceeds a pre-defined threshold, but the velocity-
     based residual remains small, then it is concluded that a bias in the SLAM
     measurement has occurred.
   - If both the position-based residual and the velocity-based residual exceed their
     pre-defined thresholds, then a faulty optical flow measurement is concluded,
     which could result from a frozen value or a sensor bias.

3) **Fault accommodation.** Using the fault diagnostic information, the control EKF
   reconfigures its sensor measurement vector used during the update step. If fault
diagnostic algorithm determines the SLAM signal has a frozen value or a bias in either
the x or y position, the control EKF will exclude the SLAM measurement from its output
vector. Similarly, if the optical flow reading is determined to be faulty due to a frozen
value or a sensor bias, then the algorithm will exclude the optical flow signal from the sensor measurement vector used by the control EKF for updating its state estimation.

Additionally, during each iteration if the optical flow sensor value is not updated, a counter is incremented to determine if the optical flow value is frozen. When a new optical flow value is received the counter is reset. If the counter reaches a predetermined threshold, the optical flow reading is considered frozen and is removed from the control EKFs update. This is run independently of the detection algorithm mentioned above.

In the fault detection and isolation procedure, by taking the absolute value of the position and velocity residuals and passing them through a low pass filter before comparing with the thresholds, the algorithm will have less false alarms due to quick spikes in the residual signal. The thresholds for the velocity and the position residuals can be determined based on extensive flight data collected during the nominal operating conditions to achieve a good trade-off between false alarms and missed detections.

It is worth noting that the drone by default uses the derivative of the SLAM positions as a velocity measurement for the velocity feedback loop in the x and y positional controller. If a fault in the SLAM positions measurement is determined (being frozen or having a bias), this velocity signal needs to be changed to a different source. Specifically, in the implementation running on the drone, the optical flow velocity reading is then used to close the velocity feedback instead.

When a faulty SLAM position measurement is determined, the control EKF starts estimating position based on optical flow velocity. This creates a problem because the position estimate made from the EKF based on the optical flow velocity measurement needs to be multiplied by a constant 1.4, as determined in the pre-calibration process. When excluding the SLAM signal
from its measurement vector, the position estimate of the control EKF cannot just be multiplied by the calibration factor because that would cause the position estimate to instantaneously jump, which might cause significant degradation or even instability issue in the position control loop. To mitigate this problem, the position estimate provided by the optical flow EKF, which already has the required additional calibration factor in place, is used as the feedback signal to close the position feedback loop. To make sure this happens smoothly and does not cause instantaneous changes in position, the transition happens with a rate limiter. While the difference between the rate limited switch and the optical flow position estimate is larger than a certain small threshold, the control loop will use the rate limited position estimate. Once the difference between the rate
limited switching position and the optical flow EKF position estimate is small, the control uses the optical flow EKF position estimate.

Figure 23 Sensor fault diagnosis and accommodation architecture
5 Results

5.1 Nominal Case

Figure 24 x position estimate comparison between the SLAM EKF and optical flow EKF
Figures 24 and 25 show a comparison of the optical flow EKF and SLAM EKF x and y position estimates. The y position estimates are nearly identical throughout the flight. While there is a difference between the x position estimates, it stays relatively constant throughout the flight resulting in the following position estimate residual plots.
Figure 26 x position estimate residual between the SLAM EKF and optical flow EKF

Figure 27 y position estimate residual between the SLAM EKF and optical flow EKF
Figures 26 and 27 show the residual between the SLAM EKF and optical flow EKF alongside the residual threshold. As the plots show, the residual stays below the threshold for the duration of the flight.
5.2 SLAM Freezing

To test the ability of the drone to continue flying under a frozen SLAM signal, during a flight the SLAM x position value will be artificially frozen to simulate the drone being unable to see a far wall as it is flying forward down a hallway.

Results and Analysis

Figure 28 graph comparing the x position estimates and measurements during a frozen SLAM x position condition

Figure 28 shows what happens when the SLAM reading in the x direction freezes. It is clear from the graph that the SLAM estimate blows up to very high unrealistic numbers.
While Figure 28 clearly shows that something is wrong, Figure 29 shows more clearly what is going wrong. Up until 45 seconds all the position measurements and position estimates stay close to each other. Starting from 45 seconds when the x position reading from SLAM flatlines, the SLAM EKF is not able to update its position estimate for x during the EKF update step because there are no new readings. This means that after 45 seconds the SLAM EKF x position is being estimated based on the gyroscope and accelerometer and it becomes unstable.
Figure 30 graph comparing the x position estimates and measurements during a frozen SLAM x position condition to actual drone position

Figure 30 shows how the control EKF x position compares to the actual drone position from VICON, the command signal, and the optical flow position. Throughout the flight before the fault occurs the optical flow position estimate is consistently lower than the control EKF. The fault occurs at 45 seconds and the fault detection and position control handover happens by 46.51 seconds. The initial difference in the optical flow EKF x position and control EKF x position have negligible effect on the tracking performance of the drone and enables the drone to continue flying safely.
Figure 31 graph comparing the y position estimates and measurements during a frozen SLAM x position condition to actual drone position

Figure 31 shows the y position estimates and readings. The y SLAM position is not frozen however when SLAM is disabled the control EKF will still switch to use the optical flow EKF position estimate for y. The detection and switching begins to happen at 45 seconds and completes by 46.51 seconds, respectively. The optical flow y position estimate is higher than the control EKF position estimate so when the switch happens the controlling EKF position estimate diverges from the actual drone position. Once the switch is completed the difference between the drone position and the optical flow control position estimate stays relatively constant, which means that the optical flow EKF does a good job of estimating changes in position and providing a stable and safe flight.
Figure 32 graph comparing the x velocity estimates and measurements during a frozen SLAM x position condition to actual drone velocity

Figure 32 shows that the velocity estimate coming from the SLAM EKF is also problematic because after 46.51 seconds it is being estimated using the accelerometer and gyroscope.
Figure 33 zoomed in graph comparing the x velocity estimates and measurements during a frozen SLAM x position condition to actual drone velocity

Figure 33 shows a zoomed in version of Figure 32. The SLAM EKF does a decent job of estimating the velocity up until 46 seconds. After the around 46 seconds the velocity estimate stops tracking well and then at around 50 seconds the velocity estimate goes to unrealistic negative velocities.
Figure 34 graph comparing the y velocity estimates and measurements during a frozen SLAM x position condition to actual drone velocity

Figure 34 shows the y velocity estimate from the optical flow and SLAM EKFs. The optical flow EKF tracks significantly better than the SLAM EKF however the y velocity from the SLAM EKF is not used for control, because the velocity residual does not exceed the threshold, it ultimately does not matter.

Frozen SLAM Conclusions

Based on the results shown above, the detection algorithm works to enable the drone to continue flying when SLAM stops updating values. This has also been observed when ROS crashed because the USB connection to the LIDAR mounted on the drone vibrated loose. When this happened, the drone continued to fly with the only difference being that the drone slowly
started to yaw because of gyroscopic drift. To conclude this part of the results, the algorithm was successful for detecting frozen SLAM faults and allowing the drone to continue to fly safely.
5.3 SLAM Bias

To test the ability of the drone to continue flying under a biased SLAM signal, during a flight a ramping bias will be added to the x and y SLAM position value at a flight time of 45 seconds. This is to simulate the drone being unable to see enough information in the environment to make an accurate position calculation but still providing a changing but inaccurate reading.

A biased SLAM value will cause the SLAM EKF position estimate to differ from the optical flow EKF position estimate. As long as the ramping rate of the position bias does not cause the velocity difference to exceed the threshold, the fault detection algorithm should quickly catch the difference in positions and disable the SLAM signal from the control EKF.
Results and Analysis

Figure 35 graph showing the x position comparison between the position estimates and measurements with a SLAM bias.

Figure 35 shows a comparison of all the x position measurements and estimates. The bias addition can clearly be seen starting at 30 seconds which causes the control EKF x position to start to increase until the fault is detected at approximately 32.06 seconds when SLAM is excluded from the update vector, and
control is then switched over to the optical flow EKF position estimate starting at 50.06 seconds.

Figure 36 graph showing the absolute value of the filtered difference in x position

Figure 36 shows the absolute value of the low pass filtered difference in the x position estimates from the optical flow EKF and SLAM EKF. The position estimate difference stays well below the threshold until the fault occurs at 30 seconds. At this point the average residual level increases but not enough to trigger the fault.
Figure 37 zoomed in graph showing the x position comparison between the position estimates and measurements with a SLAM bias

Figure 37 shows the same thing as Figure 35 except zoomed into the time that the fault occurs. When the fault is detected, and the positional control is switched from the control EKF to the optical flow EKF there is some loss of tracking performance during the transient but the optical flow position estimate stays closer to the actual drone position than the SLAM signal with the bias.
Figure 38 shows a comparison of all the y position measurements and estimates. The bias addition can clearly be seen starting at 45 seconds which causes the control EKF y position to increase until the fault is detected, and SLAM is
disabled from the update vector. The control is then switched over to the optical
flow EKF position estimate.

\[ \text{Absolute filtered difference in Y position comparison of optical flow and slam.} \]

Figure 39 shows the absolute value of the low pass filtered difference in the y
position estimates from the optical flow EKF and SLAM EKF. The threshold is
passed at approximately 32.06 seconds which means that the y position difference
is the first one to signal the fault. When the position threshold is exceeded the
algorithm checks the velocity threshold and, because the velocity threshold is not exceeded at this time, determines that the error is with SLAM initiating SLAM being excluded from the update vector and the transfer of control to the optical flow EKF.

Figure 40 zoomed in graph showing the y position comparison between the position estimates and measurements with a SLAM bias

Figure 40 shows the same thing as Figure 38 except zoomed into the time that the fault occurs. When the fault occurs, the positional control is switched from
the control EKF to the optical flow EKF. There is a significant difference between the optical flow y position estimate and the actual drone position. This means that when the position control is switched to the optical flow EKF there is some degradation in tracking performance although the tracking is still better than the tracking of the biased signal. Despite this degradation, the drone completes its flight safely using the detection and adaptation algorithm.

Figure 41 graph showing the x velocity comparison between the velocity estimates and measurements with a SLAM bias
Figure 41 shows the velocity estimates and readings. From the graph it is clear that the optical flow reading and the optical flow EKF velocity estimate track the VICON velocity readings very well. The SLAM EKF velocity estimate tracks the VICON velocity reading well up until around 25 seconds where there is some deviation. The SLAM EKF velocity estimate deviates again when the fault is introduced at 30 seconds and then again when the drone is switching control from 33 to 35 seconds.

![Graph showing absolute filtered difference in X velocity comparison of optical flow and SLAM.](image)

**Figure 42** graph showing the absolute value of the filtered difference in x velocity
Figure 42 shows the absolute value of the filtered x velocity difference between the SLAM EKF and the optical flow EKF. The threshold is not crossed.

Figure 43 graph showing the y velocity comparison between the velocity estimates and measurements with a SLAM bias

Figure 43 shows the velocity estimates and readings. From the graph it is clear that the optical flow reading and the optical flow EKF velocity track the VICON velocity readings very well.
Figure 44 graph showing the absolute value of the filtered difference in y velocity

Figure 44 shows the absolute value of the filtered y velocity difference between the SLAM EKF and the optical flow EKF. Similarly, the y residual only crosses the threshold after the algorithm has determined the fault and is in the process of switching position control over to the optical flow EKF.

**SLAM Bias Conclusions**

The position difference threshold is quickly exceeded while the velocity difference threshold is not exceeded until the position control is switched over
clearly indicating a bias in SLAM. Based on the results shown above, the detection algorithm works to enable the drone to continue flying when a ramping bias is added to both the x and y SLAM measurements.
5.4 Optical Flow Bias

To test the ability of the drone to detect an optical flow bias, during a flight a ramping value will be added to the y optical flow velocity signals starting at a flight time of 45 seconds. The bias was not added to the x optical flow velocity signal because the dimensions of the flight cage make it difficult for the drone to recover from an x velocity bias before it hits the net wall at the side of the cage.

A biased optical flow value will cause the optical flow EKF to have an increasingly high velocity estimate difference and will cause the position estimate difference to grow at a constant rate. If both the position and velocity thresholds are exceeded and there is a recent SLAM update, then the switching algorithm will disable the optical flow in the control EKFs update vector and as the velocity feedback loop.
Results and Analysis

Figure 45 graph showing the x position comparison between the position estimates and measurements with an optical flow bias

Figure 45 shows the x position comparison between the position measurements and estimates. The graph shows that the optical flow x position estimate drastically differs from the other x position measurements and estimates after around 62 seconds. This is significantly after the fault is injected in the y velocity and is caused by the y velocity signal reaching its maximum limit which causes it to freeze at its maximum limit. When the state space estimator sees the frozen velocity values it stops using the optical flow values to update the states causing the estimates to degrade quickly.
Figure 46 shows the absolute value of the filtered difference of the x position estimates. The fault occurs at 45 seconds and the difference crosses the position difference threshold at approximately 62 seconds. This is well after the fault has been detected and isolated using the y position and velocity residual thresholds.
Figure 47 shows a zoomed in version of Figure 45. Because the bias is injected into the y optical flow velocity signal, the fault does not affect the x position estimate until the EKF sees the maxed out optical flow signal and stops updating the velocity estimates.
Figure 48 shows the y position comparison between the position estimates and measurements with an optical flow bias.

Figure 48 shows the y position comparison between the position measurements and estimates. The graph shows that the optical flow y position estimate drastically differs from the other y position measurements and estimates shortly after the fault is injected.
Figure 49 graph showing the absolute value of the filtered difference in the y position

Figure 49 shows the absolute value of the filtered difference of the y position estimates. The fault occurs at 45 seconds and the difference crosses the position difference threshold at approximately 48.06 seconds.
Figure 50 shows a zoomed in version of figure 48. With the bias introduced being a ramping velocity bias the optical flow y position estimate quickly becomes unstable. There is some initial transient fluctuation when the fault detection and correction is happening but after this the drone returns to tracking the command signal closely.
Figure 51 graph showing the x velocity comparison between the velocity estimates and measurements with an optical flow bias.

Figure 51 shows a comparison between the x velocity measurements and estimates. The ramping bias in the optical flow reading can clearly be seen from 45 seconds to 60 seconds. Because the bias is only added to the y velocity there is not a noticeable effect until the y optical flow velocity value reaches the maximum allowed value and is frozen.
Figure 52 graph showing the absolute filtered difference in the x velocity

Figure 52 shows the absolute value of the filtered difference of the x velocity estimates. The fault occurs at 45 seconds and the difference crosses the velocity difference threshold at approximately 62 seconds once the y optical flow velocity signal reaches the maximum allowed value and the EKF is no longer updating the state off of the optical flow reading.
Figure 53 shows a comparison between the y velocity measurements and estimates. The ramping bias in the optical flow reading can clearly be seen from 45 seconds to 60 seconds. At approximately 60 seconds the velocity reading reaches the hard capped limit which causes the optical flow reading to look frozen and the EKF to stop updating and act unpredictably. The SLAM derivatives continue to track the actual drone movement throughout the flight so the velocity feedback loop is unaffected by the bias fault.
Figure 54 shows the absolute value of the filtered difference of the y velocity estimates. The fault occurs at 45 seconds and the difference crosses the velocity difference threshold at approximately 48.39 seconds. The optical flow fault detection requires that both the position and velocity difference threshold are exceeded. The y position residual exceeds the threshold at 48.06 seconds and then
the velocity exceeds the threshold at 48.386 seconds. At this point optical flow is removed from the update vector. The y velocity difference then returns back under the difference threshold at 50.964 seconds. By this point the detection algorithm has permanently removed the optical flow from the update vector because it has been in fault for too long so the residual returning below the threshold does no effect anything.

**Optical Flow Bias Conclusions**

Based on the results shown above, the bias detection and adaptation for optical flow faults works to allow the drone to continue flying even when a ramping bias is added to the optical flow velocity measurement.
5.5 Optical Flow Frozen

To test the ability of the drone to continue flying during a frozen condition for the optical flow signal, during a flight the optical flow x and y positions will be held at some value. This is to simulate the camera driver crashing or hanging so that it stops sending a camera feed to the optical flow block. This will in turn cause the optical flow value to be held at its last known value making the drone think that it has some constant velocity, assuming that it was not perfectly stationary when the value froze.

This fault is easier to detect than the bias because the control EKF will automatically stop updating the velocity estimate when the optical flow reading stops changing between each iteration. Additionally, the fault detection algorithm will see both the position and velocity difference thresholds being exceeded and disable the optical flow velocity from being available to use in the velocity feedback loop.
Results and Analysis

Figure 55 graph showing the x position comparison between the position estimates and measurements with a frozen optical flow reading

Figure 55 shows a comparison of the x position estimates and measurements. The optical flow velocity freeze happens at 45 seconds and the optical flow EKF x position estimate quickly decreases as it sees the frozen optical flow value and updates the position based on the gyroscope and accelerometer.
Figure 56 shows the absolute value of the filtered difference in the x position. The graph crosses the position difference threshold at 48.246 seconds. By the point that the optical flow EKF position starts to grow, the fault detection algorithm has already detected that the optical flow reading is frozen so the position difference crossing the threshold does not really matter, as the optical flow is already removed from the update vector of the control EKF.
Figure 57 zoomed in graph showing the x position comparison between the position estimates and measurements with a frozen optical flow reading

Figure 57 shows a zoomed in version of Figure 55 centered on when the fault occurs. By the time that the optical flow EKF position estimate starts losing stability, the fault detection algorithm will have already noticed that the optical flow value is frozen and removed it from the update vector to the control EKF. While there is a brief separation from the control EKF and the SLAM EKF, the difference is small and the control EKF quickly converges back to the SLAM position reading.
Figure 58 graph showing the y position comparison between the position estimates and measurements with a frozen optical flow reading.

Figure 58 shows a comparison of the x position estimates and measurements. Similarly to Figure 55, once the optical flow velocity is frozen the optical flow EKF loses the ability to estimate the position accurately and the estimate loses stability.
Figure 59 graph showing the absolute value of the filtered difference in the y position

Figure 59 shows the absolute value of the filtered difference in the y position. The graph crosses the position difference threshold at 48.67 seconds.
Figure 60 graph showing the y position comparison between the position estimates and measurements with a frozen optical flow reading

Figure 62 shows a zoomed in version of Figure 58 centered on when the fault occurs. There is no noticeable loss in tracking performance when the optical flow velocity reading freezes because the adaptation happens so quickly. The optical flow EKF position estimate quickly becomes unreliable after the frozen fault is injected at 45 seconds but it does not affect control because in this case the optical flow-based position estimate is not used in the control loop.
Figure 61 zoomed in graph showing the x velocity comparison between the position estimates and measurements with a frozen optical flow reading.

Figure 61 shows the x velocity comparison between the estimates and the measurements zoomed in on when the fault occurs. The optical flow velocity reading is frozen at 45 seconds causing the velocity estimate from the optical flow EKF to become unreliable. The SLAM position derivative tracks the actual drone velocity well.
Figure 62 graph showing the absolute filtered difference in the x velocity.

Figure 62 shows the absolute value of the filtered difference in the x velocity. The graph crosses the velocity difference threshold at 47.79 seconds.
Figure 63 graph showing the y velocity comparison between the estimates and the measurements zoomed into when the fault occurs. When the green line (optical flow reading) is frozen at 45 seconds the purple line (optical flow EKF velocity estimate) becomes unreliable. The derivative of SLAM (light blue) tracks the actual drone velocity well throughout the flight.
Figure 64 shows the absolute value of the filtered difference in the y velocity. The graph crosses the position difference threshold multiple times before the optical flow signals are frozen. This does not affect anything however because neither of the position difference thresholds are crossed which is required for a diagnosis. The first time that the threshold is exceeded after the optical flow values are frozen at 45 seconds is at 50.21 seconds.

**Optical Flow Frozen Conclusions**

Based on the results shown above, the frozen fault detection and adaptation for optical flow faults works to allow the drone to continue flying even when both
the x and y optical flow velocities are frozen. The fault detection algorithm timely
detects the frozen values, enabling satisfactory fault accommodation results.

5.6 Optical Flow Usability Conditions

If the optical flow camera cannot see the ground well enough to track pixel
movement, then the optical flow velocity estimate is not an accurate representation
of the drone velocity. There are a few different situations that can cause this type of
problem. Some likely possibilities include there not being enough light in the room
to see the floor well, the camera being blocked by wires or the battery, and the
drone moving too fast for the camera to track pixels between frames. While this
research does not provide any solutions to these problems, it is important to
acknowledge the limitation of the solutions and equipment discussed.

6 Conclusions and Future Research

The primary goal of this research was to implement a system that can
maintain positional stability in the presence of simultaneous localization and
mapping faults. This was accomplished by using extended Kalman filters running
in parallel to estimate the position and velocity with exclusive reference to either
SLAM position or optical flow velocity. Based on the difference of the position
and velocity estimations, a fault diagnosis scheme was developed to determine
faults in SLAM position or optical flow velocity measurements. Accordingly, a
primary extended Kalman filter used for flight control reconfigures its sensor measurement vector to provide reliable position estimate to maintain flight stability and safety. It was shown the proposed scheme was able to timely detect and isolate the frozen value fault and sensor bias for optical flow and SLAM measurements, respectively, and the flight control system quickly adapted to maintain flight safety based on the fault diagnostic information.

The current downwards facing camera on the drone has a slow sampling rate of 30 fps which means that the flying speed when using it as a velocity estimate must be very slow so that the camera does not move too far between frames for the pixels to track. Investigating how to use the full 120 fps that the camera is capable of or upgrading this with a higher refresh rate camera could greatly improve the usability of this control method in applications where faster drone movement is required.

Another disadvantage in the optical flow velocity estimation is that the camera needs to be able to see the surface below the drone clearly. Ordinarily this is not a problem, however when the drone is flying in poor lighting conditions the optical flow camera can have problems seeing well enough to track pixels. Another problematic scenario is when the drone is flying under lights that cast a changing shadow as the drone moves, which can cause the optical flow to become confused as optical flow assumes a constant brightness. One potential way of solving both
problems would be to mount a light to the bottom of the drone that shines on the area that the downward facing camera can see. This would require a source of power which could be drawn from the drone’s battery pack at the cost of reduced flight time but it could improve the reliability of the optical flow velocity estimate.

A system to maintain controllability when the slam input was frozen was successfully developed and tested. The ability of this system was not tested in a long hallway due to the immovability of the required network and computer that runs the drone control software interface. If the university could provide a mobile network and computer system, then further testing and control development could be done to fine tune the algorithm for the specific scenario of flying in a hallway.

Currently the drone sends gyroscope and accelerometer data to ROS for use in the laser match scanner odometry. It might be beneficial to investigate if the optical velocity estimate could be reported back in a similar way to improve the reliability of SLAM so that less input switching would be required in the first place. Reporting the optical flow velocity back could also make the map generated by gmapping be more reliable for path planning and obstacle avoidance. With what is currently implemented, when slam has faults and the drone switches to flying based off the velocity signals, the SLAM map will become corrupted because it will lose track of the drone position.
8 References


Difference Based Block Matching in Motion Estimation.”

_Microprocessors and Microsystems_, vol. 74, Apr. 2020. EBSCOhost,


_QUARC Targets Library_, 13 Apr. 2023,


