Prepared by Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Sophie Taylor, Guidance, Navigation and Control

Checked by Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Scott Wong, Power and Propulsion

Approved by Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Joshua Ham, Project Manager

Authorised for use by Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr Felipe Gonzalez, Project Coordinator

Queensland University of Technology

CRCSS-EESE, GPO Box 2434

Gardens Point Campus

Brisbane, Australia, 4001.

**Revision Record**

|  |  |  |  |
| --- | --- | --- | --- |
| Document Issue/Revision Status | **Description of Change** | **Date** | **Approved**  **by** |
|  | Initial Issue | 27 May 2012 | Joshua Ham |

**Table of Contents**

[**Definitions** 5](#_Toc326139226)

[1 Introduction 6](#_Toc326139227)

[1.1 Scope 6](#_Toc326139228)

[1.2 Background 6](#_Toc326139229)

[2 Reference Documents 7](#_Toc326139230)

[2.1 QUT Avionics Documents 7](#_Toc326139231)

[2.2 Non-QUT Documents 7](#_Toc326139232)

[3 Subsystem Introduction and Overview 9](#_Toc326139233)

[3.1 Guidance, Navigation and Control Requirements 9](#_Toc326139234)

[3.2 Subsystem interfaces 9](#_Toc326139235)

[3.2.1 Interface #3 – Power supply 9](#_Toc326139236)

[3.2.2 Interface #7 – Telemetry and Command 9](#_Toc326139237)

[3.2.3 Interface #8 – Propulsion commands 10](#_Toc326139238)

[3.2.4 Interface #9 – Proximity control and warning 10](#_Toc326139239)

[3.2.5 Interface #10 – Payload 10](#_Toc326139240)

[3.2.6 Interface #12 – Image processing 10](#_Toc326139241)

[3.2.7 Interface #20 – Airframe mounting 10](#_Toc326139242)

[4 Subsystem architecture 11](#_Toc326139243)

[4.1 Inertial Measurement Unit 12](#_Toc326139244)

[4.2 Guidance Computer 12](#_Toc326139245)

[4.2.1 Path planning state machine 12](#_Toc326139246)

[4.2.2 Desired velocity calculator 12](#_Toc326139247)

[4.2.3 PID controllers 12](#_Toc326139248)

[4.2.4 Survivor location store 12](#_Toc326139249)

[5 Detailed designs 13](#_Toc326139250)

[5.1 IMU 13](#_Toc326139251)

[5.1.1 Hardware 13](#_Toc326139252)

[5.1.2 Extended Kalman Filter 13](#_Toc326139253)

[5.2 Guidance computer 15](#_Toc326139254)

[5.2.1 Hardware 15](#_Toc326139255)

[5.2.2 Autopilot 15](#_Toc326139256)

[6 Conclusion 17](#_Toc326139257)

[7 Appendix A – Kalman filter papers 18](#_Toc326139258)

[8 Appendix B – MPU-6000 datasheet extract 19](#_Toc326139259)

[9 Appendix C – Honeywell HMC6883L datasheet extract 20](#_Toc326139260)

[10 Appendix D – ATMega 328 datasheet extract 21](#_Toc326139261)

[11 Appendix E – BeagleBone rev A5 datasheet extract 22](#_Toc326139262)

[12 Appendix F – TI AM3358 processor datasheet extract 23](#_Toc326139263)

**Definitions**

|  |  |
| --- | --- |
| QUT | Queensland University of Technology |
| TS | Trade Study |
| UAV | Unmanned Aerial Vehicle |
| GNC | Guidance, Navigation and Control |
| Circumnavigation | Flying around the disaster zone. |
| Detect  Obstacles  RD  IMU | To discover or determine the presence of something whether it be a wall, table or person.  The walls, floor or ceiling.  Reference Document  Inertial Measurement Unit |
| GC | Guidance Computer |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Introduction

The design of the Guidance, Navigation and Control subsystem for the Zephyr Group 4 Search and Rescue UAV is contained in this design document. It is expected to undergo several iterations before settling on a final design, as integration and testing reveals flaws. Thus, this initial design, Issue 1.0, is a preliminary design.

Since the software design is still at a relatively high level of design, it will become rather more detailed in following revisions of this document.

## Scope

The GNC subsystem will be introduced by a description of how it fits in to the overall system, and the design of the subsystem will be outlined. Hardware designs will be shown via referencing external technical documents, whereas software and algorithmic designs will be shown using flow charts, state machine diagrams, and psuedocode. The rationale for components of the design will be provided where appropriate.

## Background

Student teams from ENB354 are required to design, build, test and demonstrate a dirigible search and rescue platform for ARCAA. The blimp is required to navigate autonomously around a disaster zone and seek and identify survivors. If required, the blimp must be able to deliver a rescue package to a stranded survivor. The platform is controlled via a remote ground control station, where platform telemetry and imagery is displayed, and operator commands are issued. Further information on the project can be found on the project brief document which can be found on ENB354’s university database.

# Reference Documents

## QUT Avionics Documents

|  |  |  |
| --- | --- | --- |
| RD/1 | Customer Needs Document | The ZEPHYR Project Indoor Search & Rescue 2012 |
| RD/2 | ZEPH4-PMP-2012-0001 | ZEPHYR 2012, System Requirements |
| RD/3 | ZEPH4-GNC-TS-01 | ZEPHYR 2012 Guidance, Navigation and Control Trade Study for the Guidance Computer and Inertial Measurement Unit |
|  |  |  |

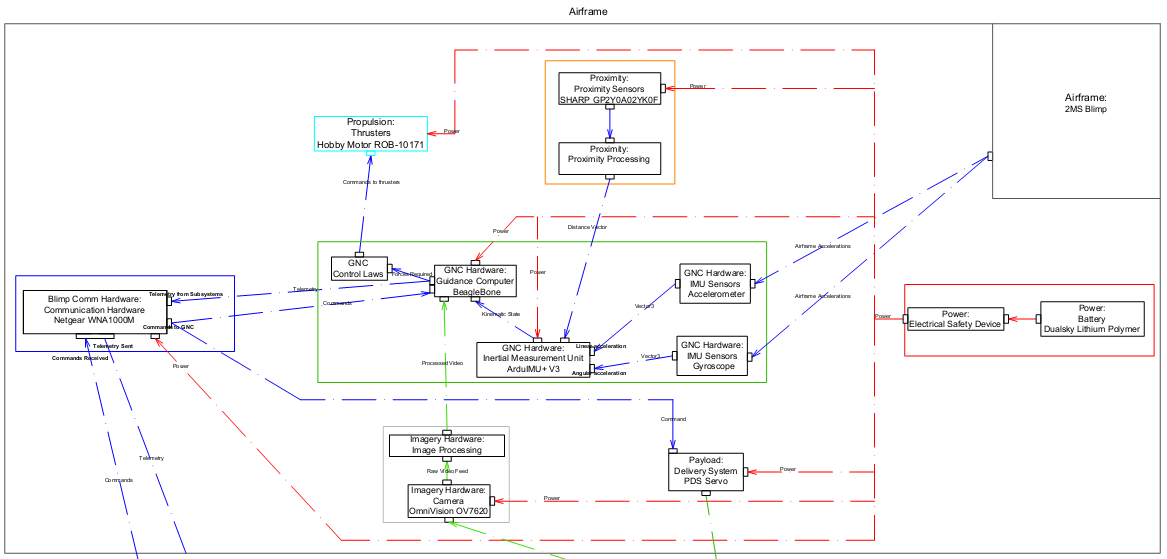
## Non-QUT Documents

|  |  |  |
| --- | --- | --- |
| ERD/1 | Avionics Navigation Systems | M. Kayton and W. Fried, Avionics Navigation Systems, 2nd ed., Wiley Interscience, 1997 |
| ERD/2 | Motion Measurement using Inertial Sensors, Ultrasonic Sensors, and Magnetometers with Extended Kalman Filter for Data Fusion | He Zhao; Zheyao Wang; , "Motion Measurement Using Inertial Sensors, Ultrasonic Sensors, and Magnetometers With Extended Kalman Filter for Data Fusion," *Sensors Journal, IEEE* , vol.12, no.5, pp.943-953, May 2012 doi: 10.1109/JSEN.2011.2166066 |
| ERD/3 | ArduIMU+ V3 schematic | <http://stuff.storediydrones.com/ArduIMU328_V3_Eagles.zip>, accessed 27/05/12 |
| ERD/4 | Blimp path planning | Y. Bestaoui, S. Hima 'Some insights in path planning of small autonomous blimps' Archives of control sciences, vol 11,2001, pp. 139 - 166. |
| ERD/5 | Blimp path planning | M. Acanfora, “New approach and results on the stability and control of airship”, UNINA |
| ERD/6 | Various parts of Control Theory | K. Ogata, “Modern Control Engineering”, 4th ed. Prentice-Hall electrical engineering series. Prentice Hall, 2002 |

# Subsystem Introduction and Overview

The Guidance, Navigation and Control subsystem’s purpose is to fly the blimp. It has several functions, captured in the requirements (Section 3.1), detailing its purpose. Put simply, they are: determine where the blimp is, where to fly to, and how to get there. The scope of the GNC subsystem means that it is mostly a software-based subsystem.

In the following diagram, the GNC subsystem is contained within the green box. As it is the overall control subsystem for the blimp, it interfaces with almost every other subsystem. Thus, the interfaces are an important part of the design, and are contained in this document, as well as the interface control document.



## Guidance, Navigation and Control Requirements

The requirements for this subsystem are the followings (RD/2):

Table 3-1 2012 ZEPHYR Group 4 Guidance, Navigation and Control Requirements

|  |  |
| --- | --- |
| Subsystem Requirements | Description |
| Functional Requirements | |
| REQ-Q | The Guidance, Navigation and Control subsystem shall determine the position of the airborne segment to within 25cm. |
| REQ-R | Airframe propulsion control commands shall be generated by the Guidance, Navigation and Control subsystem. |
| REQ-S | The Guidance, Navigation and Control subsystem shall provide control commands to the propulsion subsystem in order to move the airborne segment to a specified position. |
| REQ-T | The Guidance, Navigation and Control subsystem shall prevent the airborne segment leaving the disaster zone. |

## Subsystem interfaces

The reasons for these interfaces will become apparent in the sections detailing the design.

### Interface #3 – Power supply

The GNC subsystem requires power at 5 volts, in the form of either a DC barrel jack, or a pair of wires. The subsystem may draw up to 5 Watts of power; however, this may be lowered in later revisions.

### Interface #7 – Telemetry and Command

Due to the complexity of the subsystem, and requirements for the communications subsystem, telemetry needs to be generated. This includes kinematic state data, which is composed of position, attitude, velocity, and angular velocity; the current waypoint; the operating mode; and the thrust and torque required from the propulsion system.

The subsystem must also accept commands from the operator, via the ground control station and communications interface. Example commands are travelling to a specific waypoint, or a change of operating mode.

### Interface #8 – Propulsion commands

One of the main functions of the GNC subsystem is to provide commands to the propulsion subsystem. This will take place in the form of two vectors; one for linear thrust, and one for torque. In return, it requires the actual thrust and torque generated, in order to feed back into the control system.

### Interface #9 – Proximity control and warning

Since one of the requirements is not to collide with any obstacles, the proximity control subsystem sends a warning to the GNC subsystem when a collision is imminent, in the form of a vector to the approximate direction of the obstacle, and hands control over to the GNC subsystem in the form of a processor interrupt.

### Interface #10 – Payload

The GNC subsystem informs the payload subsystem when it is in position to deliver the rescue package to the survivor.

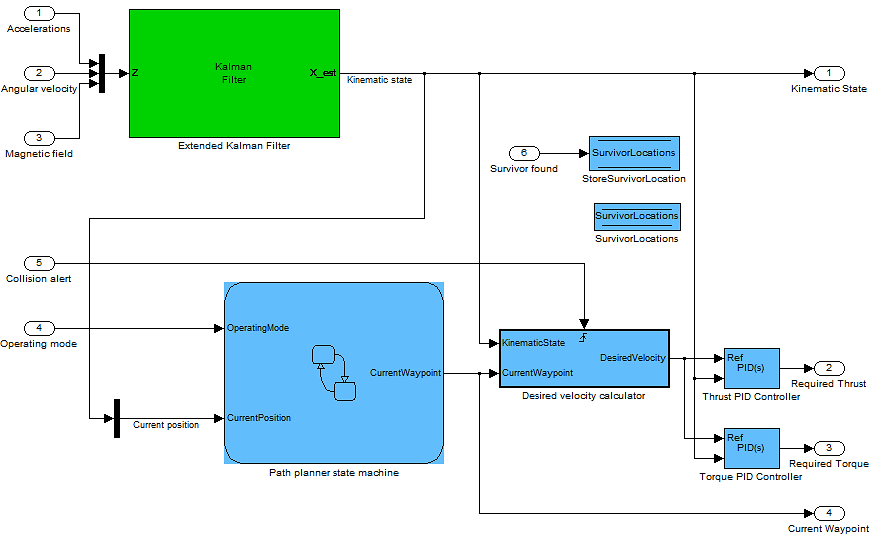
### Interface #12 – Image processing

The image processing subsystem informs the GNC subsystem of the location of any discovered survivors, so it can store them for when payload delivery takes place.

### Interface #20 – Airframe mounting

The two main components of the GNC subsystem will be placed separately. The guidance computer, not being sensitive to the vibrations from the propulsion subsystem or airframe sway, will be placed in the gondola without any padding. The inertial measurement unit, however, will be strapped directly to the airframe as close to the centre of mass as possible. It will also be cushioned with foam padding to dampen out the vibrations from the propulsion subsystem.

# Subsystem architecture



## Inertial Measurement Unit

The Inertial Measurement Unit is represented by the green Extended Kalman Filter box, as well as input ports 1, 2 and 3. This component estimates the kinematic state of the blimp, fulfilling REQ-Q. It is a component that is physically separated from the rest of the subsystem, on its own circuit board.

## Guidance Computer

The components of the guidance computer are represented by the blue boxes and the rest of the input and output ports.

### Path planning state machine

The path planning state machine is a finite state machine that generates waypoints based on the current operating mode. For example, in wall following mode, it generates waypoints that circumnavigate the disaster zone, whereas the searching mode generates a grid of waypoints.

### Desired velocity calculator

The desired velocity calculator takes in the current kinematic state and the current waypoint, and calculates the optimal speed and heading to reach the waypoint, taking into account the aerodynamic model of the blimp. For example, it takes into account the minimum turning radius and the inertia.

### PID controllers

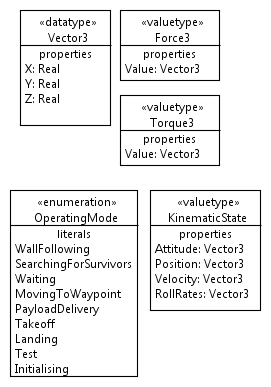
The PID controllers take the desired velocity, and compute thrust and torque settings for the propulsion subsystem.

### Survivor location store

The survivor location store is a simple set of variables that get written with the location provided from the imaging subsystem.

# Detailed designs

Common to both major components are several classes. In the following diagram, “Real” denotes a IEEE-754 64-bit float.



## IMU

### Hardware

The IMU component is an ArduIMU+ V3, selected via trade study (RD/3). It includes an integrated 3 axis accelerometer and 3 axis gyroscope (MPU-6000, Appendix B), a 3 axis magnetometer (HMC-5883L, Appendix C), and a small microcontroller (ATMega328, Appendix D). It connects to the guidance computer via a UART connection.

### Extended Kalman Filter

(See ERD/1 and ERD/2 for more complete descriptions)

The Kalman filter is a recursive algorithm for linear state estimation. By taking into account noise from both a plant and a measurement system, it takes a stream of noisy data and produces an optimal estimate of the underlying plant state. This means that a more accurate knowledge of the plant state is possible than a naïve plant model. The original paper describing the Kalman paper can be found in Appendix A.

The extended Kalman filter, or EKF, is a nonlinear extension to the Kalman filter, which linearises the plant model about the current mean and covariance. This produces more accurate estimates for nonlinear systems, such as an aerodynamic model. However, it is a first order estimate, as it ignores higher order terms in the Taylor expansion of the plant model while linearising.

The EKF used in the IMU is of the form described in ERD/2; a copy of which can be found in Appendix A. The noise covariance data can be found in the data sheets for each sensor (Appendix B and C).

The EKF runs on the ATMega328.

## Guidance computer

### Hardware

The guidance computer is a BeagleBone rev. A5 (Appendix E), selected via trade study (RD/3). On it runs the Angstrom Linux distribution as a high level environment to program against.

### Autopilot

The main program that runs on the guidance computer, the Autopilot program is what contains the remaining components. It will be implemented in C++, with some of the C++ code generated by Simulink.

Psuedocode:

Init:

1. Initialise IMU;
2. Initialise path planner;
3. Initialise SurvivorLocations store;

Loop:

1. Get kinematic state from IMU, add to telemetry structure;
2. Check for new operating mode; pass on to path planner;
3. Get waypoint from path planner, add to telemetry;
4. Calculate the desired velocity from the aerodynamic model;
5. Update PID controllers; get required thrust and torque;
6. Send thrust and torque to propulsion subsystem program
7. Pass telemetry to telemetry program

#### IMU communication

As the IMU outputs over a serial UART connection, the Autopilot program must read from the BeagleBone UART port to extract the kinematic state values. As this is just a single wire running from the TX pin from the IMU to one of the RX pins on the GC, no drawing is provided. It then forms the KinematicState structure based on the output.

#### Path planning state machine

The path planning state machine is an extended finite state machine. The current state depends on the operating mode selected by the operator. Each state generates waypoints in a different method.

##### Wall following state

The waypoints are generated via interpolation of the dimensions of the room.

##### Searching for survivors state

The waypoints are generated in a grid, with the next waypoint chosen to be either in a spiral pattern, or a back-and-forth pattern.

##### Moving to waypoint state

The waypoint is set via an external command from the operator, via the ground control station and communication subsystem.

##### Payload delivery state

The survivor to deliver the payload to is chosen by the operator on the ground control station, which notifies the path planner, which waypoint to read from the SurvivorLocations memory store.

#### Desired velocity calculator

The desired velocity calculator takes in the current kinematic state and the current waypoint, and calculates the optimal speed and heading to reach the waypoint, taking into account the aerodynamic model of the blimp. By using a 6 degree of freedom model, desired trajectories can be put in, and the model is solved for the velocities and forces. ERD/4 and ERD/5 explain this in further detail, and the subsystem will be using the methods within.

#### PID controllers

PID controllers take in an error value, and output a value that is a weighted sum of the error input, the derivative of the error input, and the integral of the error input. This flexible scheme allows many complicated plants to be controlled easily. ERD/6 details the mathematical theory of PID controllers and their design.

#### SurvivorLocations memory store

This is a simple memory array, with space for 3 waypoints to be stored.

Example:

Vector3[3] SurvivorLocations = {Vector3(), Vector3(), Vector3()};

# Conclusion

By combining a System on a Chip computer with MEMS sensors, efficient autopilots with reasonable accuracy can be achieved at low weight, low power usage, and low price. By using a full computer and OS, powerful algorithms can be run, with the ease of debugging on desktop PCs.

# Appendix A – Kalman filter papers

# Appendix B – MPU-6000 datasheet extract

# Appendix C – Honeywell HMC6883L datasheet extract

# Appendix D – ATMega 328 datasheet extract

# Appendix E – BeagleBone rev A5 datasheet extract

# Appendix F – TI AM3358 processor datasheet extract