Software Requirements Specification

for

Autonomous Quadcopter

Prepared by:

Lakshya Sharma (101683039)

Ishaan Kohli (101503097)

Ishita Sharma (101503100)

Gaurav Dua (101503072)



Thapar University

May, 2018

Table of Contents

Table of Contents ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. Other Nonfunctional Requirements 4

4.1 Performance Requirements 4

4.2 Safety Requirements 5

4.3 Security Requirements 5

5.  **Feasibility ……………………………………………………………………………………..6**

5.1 Operational Feasibility………………………………………………….………………………….6

5.2 Technical Feasibility...………………………………………………….………………………….6

5.3 Economical Feasibility………………………………………………….………………………….6

5.4 Legal Feasibility……..………………………………………………….………………………….6

6. Other Requirements 5

Appendix A: Analysis Models 5

# Introduction

## Purpose

The purpose of this project is to develop Semi-Autonomous Quadcopter. This UAV will be able to connect to the google maps online api to find a way between two GPS locations and be able to follow that path to reach the destination. Also, to counter the obstacles it finds during its flight, there would be a suitable algorithm which helps it to avoid those obstacles and re route it’s way accordingly to have a successful flight.

## Document Conventions

The font used during the SRS was Times New Roman. For normal text, the font size was 12, and 14 for headings. The headings were in bold. The text was left indented.

## Intended Audience and Reading Suggestions

The intended audience for this document would be the students who want to learn more about this project or want to use it as a reference for their own project.

## Product Scope

The scope of this technology would be future students who are doing a similar project. Additionally, it would be for people who would want to automate their day to day task by using a quadcopter.

# Overall Description

## Product Perspective

This product is a new, self-contained product.

As an intuitive capstone project, we wanted to do something which would require give us a fruitful experience to enter the world of making project. There are various ways in which you can automate a quadcopter, for example self-flight, security UAV etc. Our autonomous quadcopter would be able to move through two Given GPS coordinates by itself and also be able to avoid the given obstacles on its own.

The basic use of such a system is sending any kind of item from one point to another, basically as a delivery drone. This can help a user in day to day task like for watering plants, spraying insectescide on the crops.

The User will have to feed two GPS coordinates and a desired height and the quadcopter will automatically move in that direction. Then it would take off and land on the respective gps coordinates. The same is given in the Use Case diagram(See Appendix A)

## Product Function

The product performs the following functions:

* Get the GPS coordinates from the user using the computer software interface.
* Get the Height from the user.
* Take-off from the starting point, and land on the desired final gps coordinate
* Would be able to detect obstacles and avoid them.
* To have a stop command which would immediately stop the Quadcoper and land it safely on the same location.

## Operating Environment

This is an embedded system, so there are no constraints with respect to the running environment. However, the microcontroller that we would be using is the Arduino Uno and use the Ardu Pilot flight controller board. All the sensors are processed in FMS itself.

The software on the computer would be able to run on windows only since we are developing the software for that platform.

## Design and Implementation Constraints

The design constraints that we faced where the dimensions of the Quadcopter frame, and various other factors like centre of gravity, weight etc for a smooth flight. We have used the F450 quadcopter frame to build our quadcopter.

The only legal constraint that we can face is the restriction of UAV in India. The maximum height that we can test our quadcopter is 200m and that would be kept in mind while designing the code.

## User Documentation

We referred to Stack Overflow (*https://www.stackoverflow.com/*) for looking up for algorithms and ciruits.io(<https://circuits.io>) for simulations and debugging.

## Assumptions and Dependencies

We have assumed that the flight environment will not be turbulent while designing the algorithm. We are dependent for routes on the google maps api.

We are also going to use the laptop for image processing and making judgements about obstacles, and hence there is an active network connection between the Camera and the PC.

# Interface

## User Interfaces

The User interface will be very simple. It would only include the 2 options, first would be entering the GPS Coordinates, and second would be setting the height. The same would be developed as an Application or a web interface.

## Hardware Interfaces

The hardware interface would consist of an Arduino Uno board which would help us to interface the various sensors, which are Gyroscopic, Accelerometric, GPS, Barometric sensors to find the correct orientation of the Quadcopter and also to correctly stabilize the quadcopter when left in idle state. The Arduino official software would be used to upload the code onto the flight management sensor.

## Software Interfaces

The software interfaces that we would be using would mainly include Google Maps Api to find the route between the Start and the destination and the other interfaces that we would be including are ArduPilot and MultiWii which are open source projects that are specifically designed for the better stabilization of the Quadcopter and make changes to the quadcopter to our need.

## Communications Interfaces

The communication interfaces would include a Bluetooth module and a wifi module. The commands would be sent by both the Bluetooth and the wifi module. However, the wifi module would also be specifically designed for network communication for attaining correct user data.

# Other Nonfunctional Requirements

## Performance Requirements

The system must be user-friendly and the delays involved must be less. So in every action-response of the system, there are no immediate delays. In case of opening forms, of popping error messages and saving the settings or sessions there are no noticeable delays.

## Safety Requirements

Information transmission should be securely transmitted to server without any changes in information.

## Security Requirements

The main safety requirement for this project is that mid-flight, there should be no way that there can be sent a command that can alter the flight route or any behavior that the UAV must not show.

# Feasibility

## Operational Feasibility

The quadcopter is an example of an embedded system. So, the first thing in operational feasibility is that how good an embedded system is a quadcopter. The main operation of the quadcopter is flying. The flying can be easily achieved using an Arduino or any other microcontroller.

The other operations that are also the main functional requirements of the Project are: -

1. GPS point to point movement
2. Detection of Obstacles using camera

The completed project would consist of the above two actions resulting in the flying of the quadcopter, or one can say that these actions would in turn give commands to the quadcopter to fly according and adjust its course. With due help of various research papers, it has been found that this is possible. The first thing, GPS movement can be implemented, and secondly, detecting object can also be done using a camera or an Ultra Sonic Sensor. The challenge is to combine these two and find whether which input is more favorable at a point of time and making decision. It has been seen that it is possible to implement such an algorithm in which, whenever an obstacle is detected, the quadcopter first readjusts itself to avoid the obstacle and then it corrects its orientation and then continue its course from that point. This can easily be implemented and is operationally feasible and hence can be implemented.

## Technical Feasibility

The technical feasibility can be divided into parts:

### Hardware Feasibility

The hardware required for this project is a microcontroller, a gyroscope and accelerometer sensor, a gps sensor, a wifi module and a Bluetooth module. All this will comprise of the Flight management system or the flight controller.

The other parts that are required are 4 brushless motors, 4 ESC (Electronic Speed Controllers), rotors and a quadcopter frame and a battery.

All these parts are readily available and are also in the process of being ordered. The connections are also readily available on the internet for use.

### Software Feasibility

The help from three things would be required, Python-For image processing, Arduino(C/C++) for controlling the Arduino and Html, CSS and a backend to write a web interface.

For proper functioning, the interface and image recognition should be able to send commands over to the Arduino board so that the Arduino can take necessary steps. This is possible with the help of libraries found on Pypi, and NPM and can be done easiliy.

## Economic Feasibility

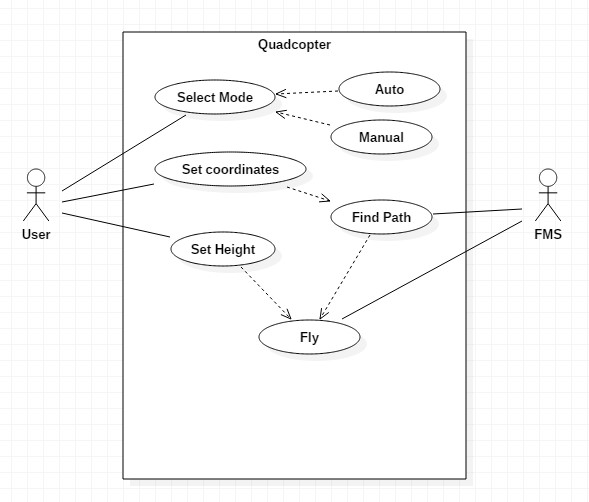
All in all, the project would cost around Rs. 10000 which is not very much and is being provided by the college and hence is feasible economically.

## Legal Feasibility

The project is Legally feasible in other countries, but in India, there is a restriction of height that the quadcopter can reach and hence the software would be designed such that restriction is not Crossed.

**Appendix A-Analysis Models**

1. Use Case Diagram



1. Use Case Templates

|  |  |
| --- | --- |
| Use Case ID: | 1 |
| Use Case Name: | Select Mode |

|  |  |
| --- | --- |
| Actor: | User |
| Description: | Select the mode in which the quadcopter has to fly (auto/manual) |
| Preconditions: | - |
| Postconditions: | - |
| Priority: | High |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | First event |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | Mode A (manual), Mode B (Auto) |
| Special Requirements: | - |
| Assumptions: | The quadcopter has just started and is waiting for first instructions from user |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| Use Case ID: | 2 |
| Use Case Name: | Set Coordinates |

|  |  |
| --- | --- |
| Actor: | User |
| Description: | Set the latitude and longitude of destination |
| Preconditions: | Mode should be auto |
| Postconditions: | - |
| Priority: | High |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | The Quadcopter has been set to auto mode and is waiting for instructions from user |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| Use Case ID: | 3 |
| Use Case Name: | Set Height |

|  |  |
| --- | --- |
| Actor: | User |
| Description: | Select the height at which the quadcopter has to fly (max 3m) |
| Preconditions: | - |
| Postconditions: | - |
| Priority: | Low |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates -> Set Height |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | - |
| Assumptions: | - |
| Notes and Issues: | - |

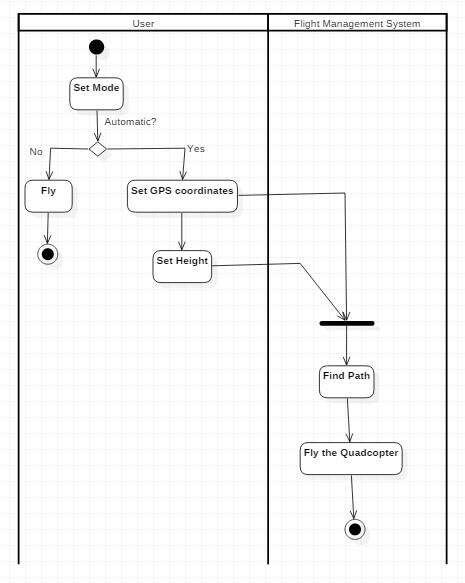
|  |  |
| --- | --- |
| Use Case ID: | 4 |
| Use Case Name: | Integrate with Google Maps |

|  |  |
| --- | --- |
| Actor: | Flight Management System |
| Description: | Find path which it has to follow |
| Preconditions: | Coordinates must have been set and mode should be auto |
| Postconditions: | - |
| Priority: | Medium |
| Frequency of Use: | Once in a flight |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates -> Set Height (Optional) -> Integrate with Google Maps |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | Coordinates must have been set and mode should be auto |
| Assumptions: | - |
| Notes and Issues: | - |

|  |  |
| --- | --- |
| Use Case ID: | 5 |
| Use Case Name: | Fly |

|  |  |
| --- | --- |
| Actor: | Flight Management System |
| Description: | Fly with the given instructions |
| Preconditions: | Coordinates must have been set if mode is auto |
| Postconditions: | - |
| Priority: | High |
| Frequency of Use: | Continuous |
| Normal Course of Events: | Mode(Auto) -> Set Coordinates -> Set Height (Optional) -> Integrate with Google Maps -> Fly |
| Alternative Courses: | Mode (manual) -> Fly |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | Coordinates must have been set if mode is auto |
| Assumptions: | Default height is 3m |
| Notes and Issues: | - |

1. **Activity/Swimlane Diagram**



1. **Sequence Diagram**

