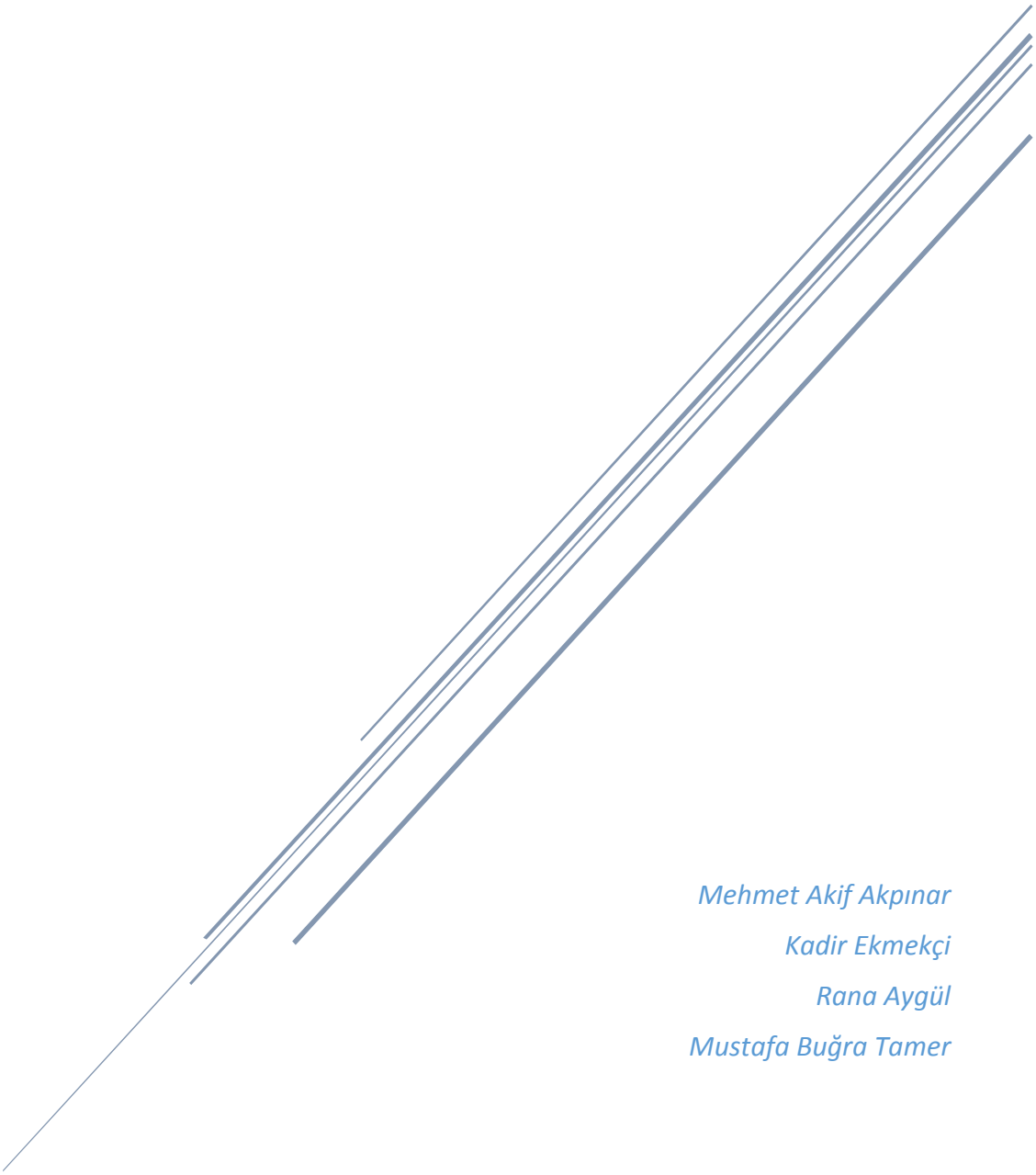
RAVEN NEST

CENG 490 Term Project



Mehmet Akif Akpınar

Kadir Ekmekçi

Rana Aygül

Mustafa Buğra Tamer

Middle East Technical University
Computer Engineering Department

Table of Contents

1 OVERVIEW

The Software Design Document gives extensive information about how the software and hardware should be built and how their interactions flow along the project. This design report includes a complete design description about Raven Nest project. It also contains the features, functionalities, specifications and explanations about project. Details and related interactions for the system are represented using graphical notations such that sequence diagrams, use case models, state diagram etc.

1.1 Scope

In this design document, all system components and a comprehensive design descriptions for each model are explained. There will be a set of design views that will contribute to specify the design and development process and to clarify the overall project steps and operations. The document provides a clear understanding for the main structure of the project and how the implementation process will be carried out.

1.2 Purpose

This software design document provides the system modeling and architectural design for Raven Nest project. The main purpose to prepare this document is to give a general description for what the design elements are and how they interact with each other, what the system components and behaviors are and how they have a significance on the system operation. It is prepared to make primary reference to the implementation phase of the project. Moreover, this document verifies whether the design topics conform to the requirements specified in SRS document.

1.3 Intended Auidence

This System Design Description document targets to different audience for different purposes. First of all, the main audience for this document is the members of the project group, system developers and advisors of this staff. Since this project is the combination of hardware and software, it addresses to developers and testers that are going to develop the system software while implementing system requirements. Moreover, since the usage fields of this system is not limited with research areas, it is used in industrial area and it is a comprehensive document to understand how the system operations work and what the main components that are composed of this system are. Thus, it also addresses to end users and researcher interested in this project topic. For validation and verification of test phase, it is a helpful reference to the testers to find and fix the bugs on the implementation phase.

2 DEFINITIONS

TERMS	DEFINITIONS
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
Quad-copter	A multi-rotor helicopter that is lifted and propelled by four rotors
AR Drone	A radio controlled flying quadrocopter helicopter.
ROS	Robot Operating System
OpenCV	A library of programming functions mainly aimed at real-time computer vision
IEEE	Institute of Electrical and Electronics Engineering
Gazebo	Robot simulation tool
SDD	System Design Document
StarUML	An open source Unified Modeling Language tool
AI	Artificial Intelligence
SRS	Software Requirements Specification
tum_simulator	Gazebo simulator package for AR Drone
tum_ardrone	ARDrone implementation package

3 CONCEPTUAL MODEL FOR SOFTWARE DESİGN DESCRİPTİONS

This section contains basic project terms, definitions and the context of SDD that the document is prepared. The purpose of this section is to give a comprehensive descriptions for project terminology, software life cycle and the importance of validation and verification of design.

3.1 Software Design in Context

This project will be designed as three main subsystem for multi-scenario in modular fashion. Thus, the project shall be reliable, costly efficient, quickly responsive to user's demands and stable in both real-time environment and simulation environment. Moreover, since this project is based on a real-time robotic system, it is required that robots shall carry out the specified scenarios in the stable way. However, since we are working with UAV and UGV and interested in with their interactions, reliability and stability are a critical issue for us. For the correctness of artificial intelligence algorithms that we will implement for both vehicle and for complete feasibility of scenarios, auto-pilot application that we coded for UAV shall be stable, give precise results for required points and follow the created paths in the correct way after algorithm is applied. Moreover, some artificial intelligence algorithms for path planning, OpenCV library for image processing and C++ programming language with OpenGL library will be used during development of the project.

3.2 Software Design Descriptions within Life Cycle

3.2.1 Influences on SDD Preparation

This document is prepared by considering the opinions of stakeholder which is Assc. Prof. Selim Temizer and requirements specified in SRS document. The SRS document will be a significant reference for preparing this document. However, this document can be influenced by additional requirements or design choices that can occur during design process.

3.2.2 Influences on Software Life Cycle Products

This project is a project that is composed hardware and software parts,has artificial intelligence and computer vision algorithms and that uses different robots with multi-scenario such as AR Drone 2.0 as UAV and P3DX as UGV. The first goal that shall be achieved is to prepare all functionalities of the project on simulation environment before passing to real-time operations. Since the project is an academic-based project, we are responsible for our advisor Selim Temizer. We develop the project according to his feedbacks and our researches based on academic studying.

Moreover, we shall know that the system is ready for auto-pilot usage before running artificial intelligence and computer vision algorithms. Therefore, the first thing that we should carry out is to hover the UAV with auto-pilot in a stable way by giving simple algorithm and paths. We aim to develop real-time simulation that is another main sub-scenario in this project and that will be progressed with the rest of the project.

3.2.3 Design Verification and Design Role in Validation

Validation and verification will be tested after all preparation of test cases and this design shall be verified to ensure that it fulfills the requirements specified in SRS document with the help of those cases. Since our project is a multi-robot operation and management system, it shall be tested in every stage as long as it is developed. Also since it is a real-time system, it shall respond quickly to commands that will be result of some algorithms and give reliable results to corresponding requirement states. Another important responsibility to achieve during the project is to simulate the real-time system on the different screen to monitor the motion behavior of the overall system. Therefore, we shall test our design and overall system constantly and concurrently throughout the development process by taking stability, reliability, efficiency and quickly response concepts into consideration.

4 DESIGN DESCRIPTION INFORMATION CONTENT

4.1 Introduction

The purpose of this part is to give a comprehensive information about how the how the design description information will be delivered in the next steps of the document. In this section we will inform users of this document about SDD identification, design stakeholders and their concerns, design views and its viewpoints, design rationale and design languages. Details about for each subsection are explained clearly below.

4.2 SDD Identification

This document is the initial SDD for the Raven Nest Project of Project Skynet. The date of issue for this document is 04.01.2015. Since there can be some changes on design and implementation part in the next processes of the project, this document is not final design document for the Raven Nest project. The issuing organization for this project is team Project Skynet which is founded by Mehmet Akif Akpınar, Rana Aygül, Kadir Ekmekçi and Mustafa Buğra Tamer. The supervisors of this project are Dr. Selim Temizer and Asst. Burak Kerim Akkuş.

The scope of this document is the software design information about project SkyNet. We will use UML for describing the design viewpoints specified in the further subsections.

4.3 Design Stakeholders and Their Concerns

The specified design stakeholders for this document are Dr.Selim Temizer, Asst. Burak Kerim Akkuş and other CENG 490 staff. Although Dr. Selim Temizer is not the instructor of CENG 490 course, he helps in every step of the project, provides us both some hardware and software stuff, leads us with his useful and important feedbacks and he is the main supervisor of team Project SkyNet. Asst. Burak Kerim Akkuş is one of the teaching assistants of the CENG 490

course and is responsible for the team Project SkyNet. The team and Asst. Burak Kerim Akkuş will have weekly meetings about how the project progress is going and what the last state of the project until that weekly meeting is. Therefore, team will have an ability to get continuous feedbacks from him and the main contact between team and course staff will be via Asst. Burak Kerim Akkuş.

4.4 Design Views

This project will be implemented by using ROS , tum_simulator and some packages for them. ROS and tum_simulator generate us a simulation environment and enable to use UAV and UGV simulation objects . These products are open source. Thus, stakeholders can also easily update the project. Product context is specified and all restricted limitations are given in the SRS document before. Team organization is equally made. Composition is as much as we can separate concepts equally. Design view also shows estimated cost, staffing, documenting and scheduling. A logical view of the product is explained more detailed in another part and also it is supported by diagrams. Relationships of the classes are easily perceived. Dependency and information viewpoints show possible future problems and how the information is stored and shared among the users. At last, state dynamic view shows the state transitions and flow of actions with diagrams.

4.5 Design Viewpoints

Context viewpoint shows what is expected from the user(how his role will be) in the system. The roles of the user and stakeholders are clearly specified. Logical viewpoint shows the logical class structure of the both UGV and UAV. Dependency viewpoint is used for explaining the dependencies among UGV,UAV and server in the system. Information viewpoint is the viewpoint which explains the structure of data about the image signals and their state over the process of solving maze . Interaction viewpoint is explaining the interactions between symbiotic autonomous robots. Finally the state dynamic viewpoint is used for explaining the maze solving in a state described way.

4.6 Design Elements

Design entities and design attributes are stated in this subsection.

4.6.1 Design Entities and Design Attributes

Design entities and design attributes are listed as following.

- OpenCV
- Image Processor
- Maze Solver
- Path Creater
- Server Interface
- Simulation Interface

4.6.2 Design Relationships

All design entities can be contained by one of the 2 subsystem of these project which are UGV or Server. Server Interface, Simulation Interface and Path Creater will be run on Server. Maze Solver, OpenCV and Image Processor will be run on UGV. All of them are critical for the finishing missions succesfuly.

4.6.3 Design Constraints

The constraints can arise from image files which should be handled properly and have the binary structure without any problem for the application to properly process the image without having a failure. Network connection must be supplied for the proper execution of the application. Apart from that, OpenCV and simulation design are the main constraints for the program.

4.7 Design Overlay

Since there is not an already-defined design view, we do not have a design overlay.

4.8 Design Rationale

Design choices are created according to some significant features like sustainability, reliability, stability, efficiency so that system can respond the required scenarios with specified cases. It can be updated according to stakeholders and users' changing requirements. Since the system that we will carry out is mostly hardware oriented and open source code is available for system components which are UAV and UGV, we are not interested in write extra code for this system except the simulation part and some algorithms for path planning and image processing. However, for each function which we will write for specified cases mentioned above we will state some comment so that user of this document can understand what the code does and how it can be integrated or altered for other similar systems when the system software is examined.

4.9 Design Languages

Unified Modeling Language (UML) is used to represent the design viewpoints listed in (4.5).

5 DESIGN VIEWPOINTS

This section will give brief knowledge about design viewpoints.

5.1 Introduction

In this part, six main design viewpoints will be explained in detail. Several design viewpoints in terms of design concerns for use will be defined.

- Context Viewpoint
- Logical Viewpoint
- Dependency Viewpoint
- Interface Viewpoint
- Interaction Viewpoint
- State Dynamics Viewpoint

During the explanation of these viewpoints, UML diagrams will be used for understanding the flow of system operations.

5.2 Context Viewpoint

It is necessary to explain the context viewpoint for the system since it interacts with user and provides functionalities. Relevant information between the actors and the services will be shown by the use case diagrams. Note that for three scenarios, same use case diagram is used but with different headers. Use cases related to different scenarios are illustrated as below:

5.2.1 Fixed Size Maze Use Case Diagram

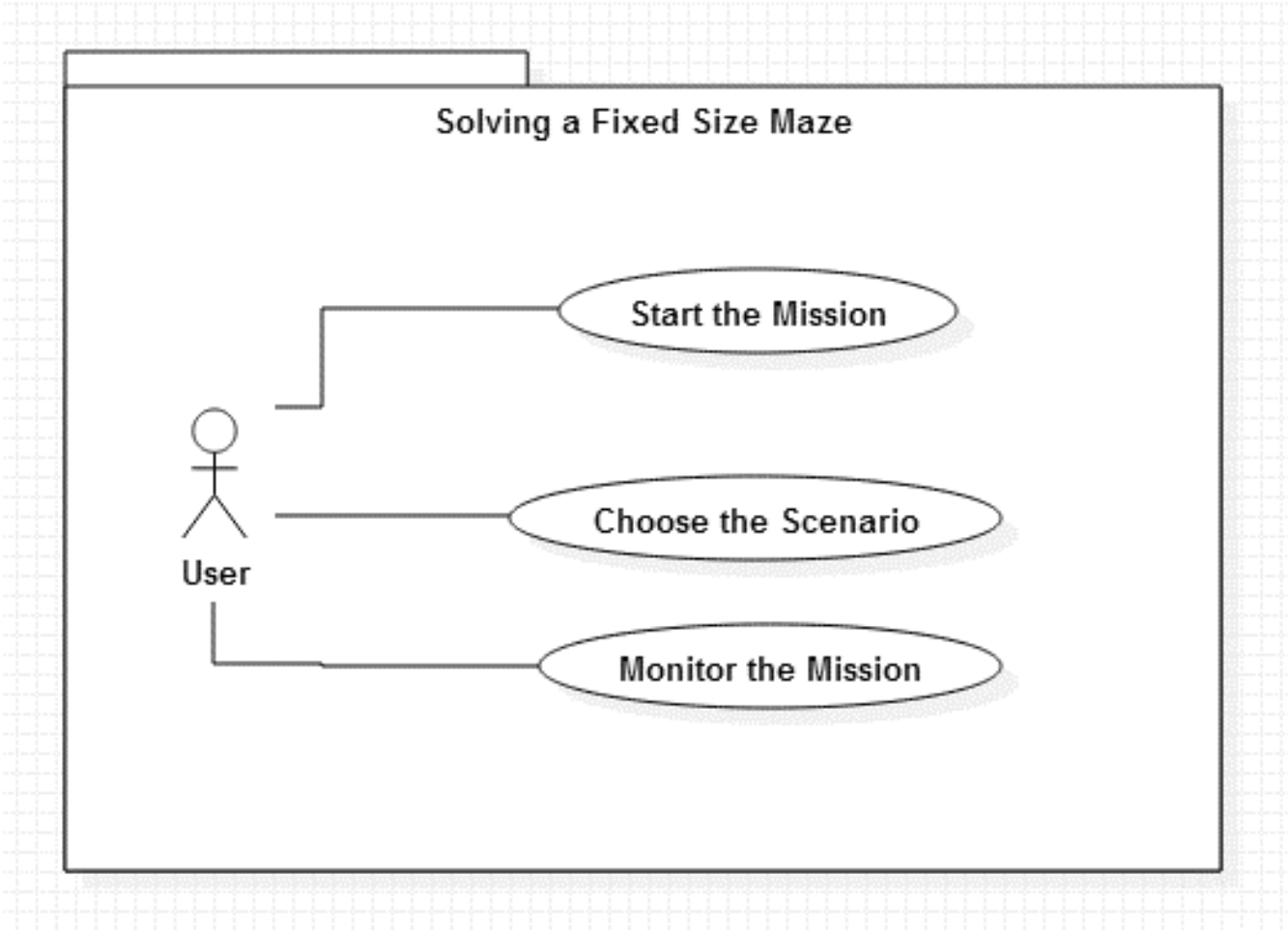


Figure 1 Use Case Diagram: Sce 1

This use case diagram will be used for describing the first scenario which is solution for a fixed size maze.

5.2.2 Unknown Size Maze Use Case Diagram

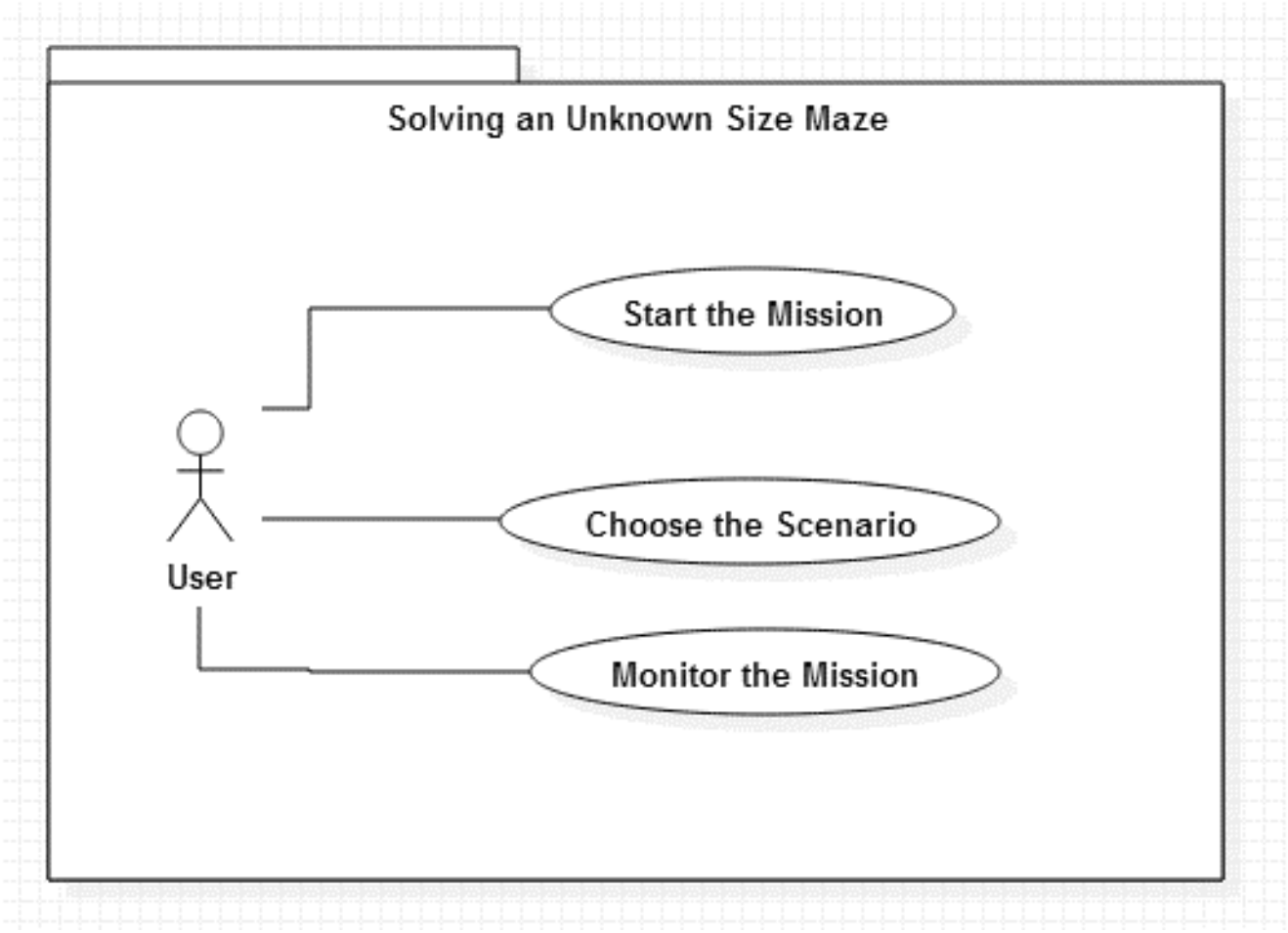


Figure 2Use Case Diagram: Sce 2

This use case diagram will be used for describing the second scenario which is solution for unknown size maze.

5.2.3 Mapping a Region Use Case Diagram

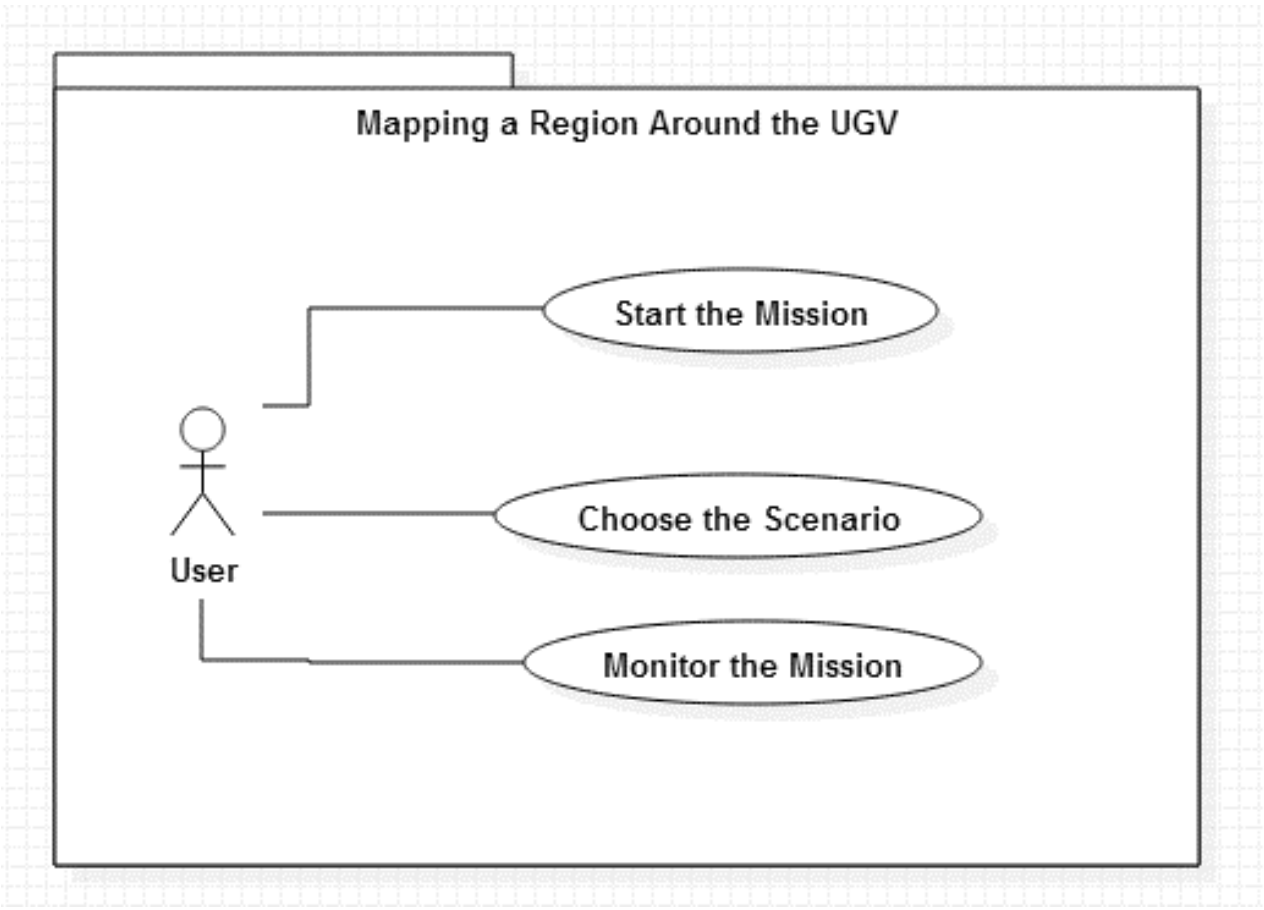


Figure 3 Use Case Diagram: Sce 3

This use case diagram will be used for describing the third scenario which is solution for mapping a region.

Since these use cases were explained in SRS document in detail, there is no requirement to explain them again. One can see part 2.2 Product functions of Raven Nest SRS document for detailed information about for use cases of specified scenarios.

5.3 Composition Viewpoint

Our system is composed of three components which are the UAG (AR DRONE 2.0), the UGV(P3-DX) and the computer, which runs Linux Ubuntu 14.04 for operating system.

More information can be found in the SRS document, 2.1.1 System interfaces subsection.

5.3.1 Example Languages

Deployment diagram of our system is shown below.

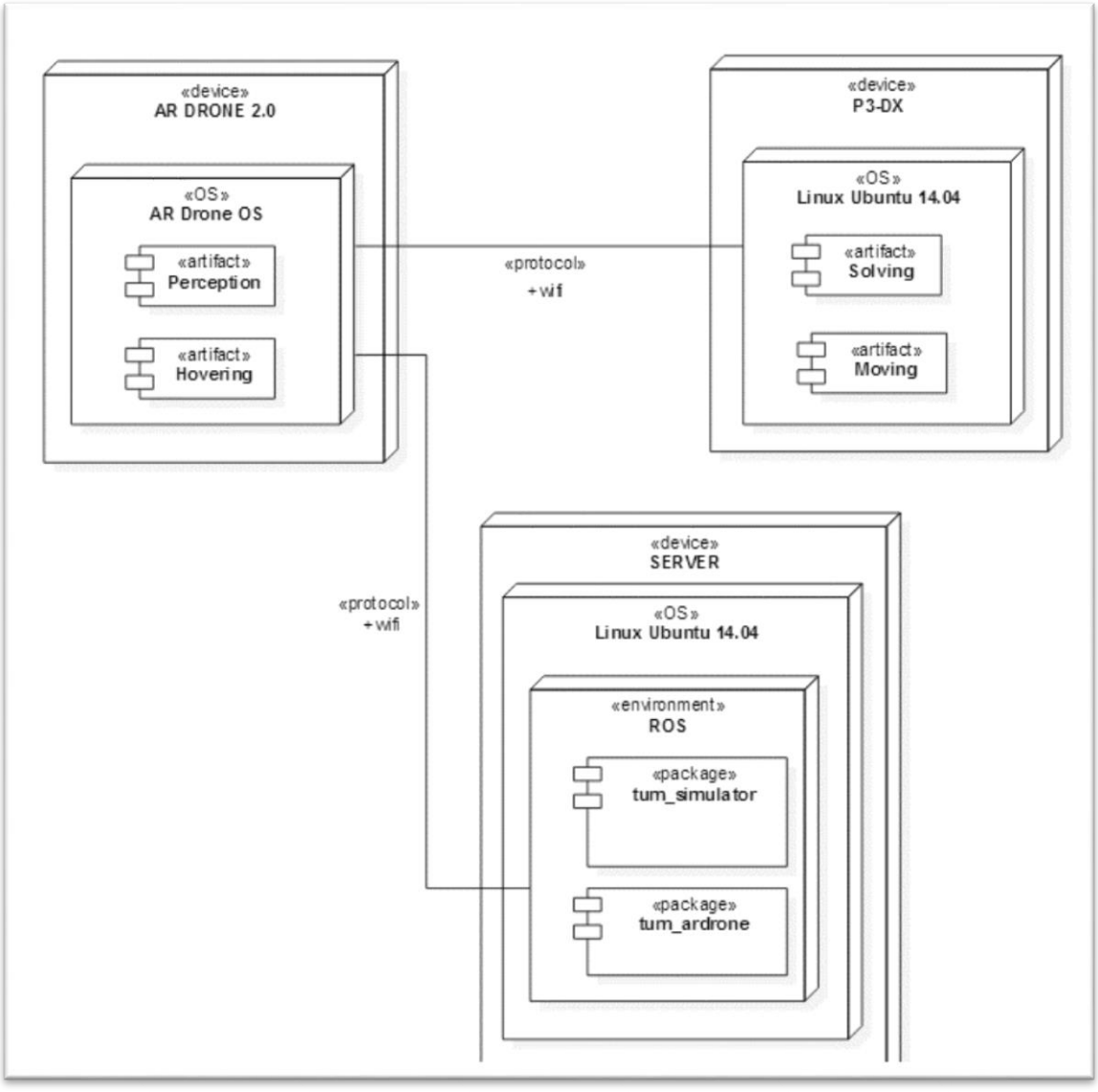


Figure 4 Deployment Diagram,

5.4 Logical Viewpoint

The logical viewpoint of the system aims to describe the implementation as classes with their structural relationships. Logical viewpoint explains the system by analyzing the classes which are constructed at the implementation step of the application.

However in our project we are not focusing any object oriented design since our project is focusing AI and CV algorithms and their implementation on robotic system and object oriented design is not necessary. Therefore there is no advanced class structure in our project.

5.5 Dependency Viewpoint

The dependency viewpoint specifies the relationships and dependencies between the design components of the system. Identifying the dependencies of the system and determining which subsystems are depends on other subsystems helps deciding the priorities in developing design entities.

In Raven Nest there are three subsystems which interacts each other which are Server, UAV and UGV. Server sends start and configuration command to UAV. UAV system will send the extracted data from the images taken to both Server and UGV. Server will use this information for creating real time simulation and UGV will use it to create the path and it moves on this path for finishing the mission.

In these processes all three system must be run correctly and synchronize but both Server and UGV are dependent to UAV. The reason for that both of them need correct information to process their facilities.

5.6 Interface Viewpoint

The system is not going to interact with humans directly when it is working, sending commands will be done in the terminal of Server. However the system provides a real time simulation which is running on the Gazebo for monitoring the robot movements.

Our algorithms, which we will start from the terminal, will write the commands ,which are needed to solve the situation at hand(maze solving, asset finding etc.) to the file, and send them to both tum_simulator and tum_ardrone. Since we will not develop a graphical user interface for our system, and use mostly terminal for interaction, the tum_simulator's and tum_ardrone's graphical user interfaces will be described in this section.

5.6.1 tum_ardrone Take Commands Screen

When you start the tum_ardrone this screen will appear. In this screen you can send commands to the AR Drone, the commands will be sent with a file ,which can be selected from the load file button. When you press the send button, the commands in the file ,that we selected before with the load file button, will be sent to the AR Drone, and AR Drone will move according to the commands ,which are in the file we sent. Also you can see the information about node communication, autopilot, and state estimation.

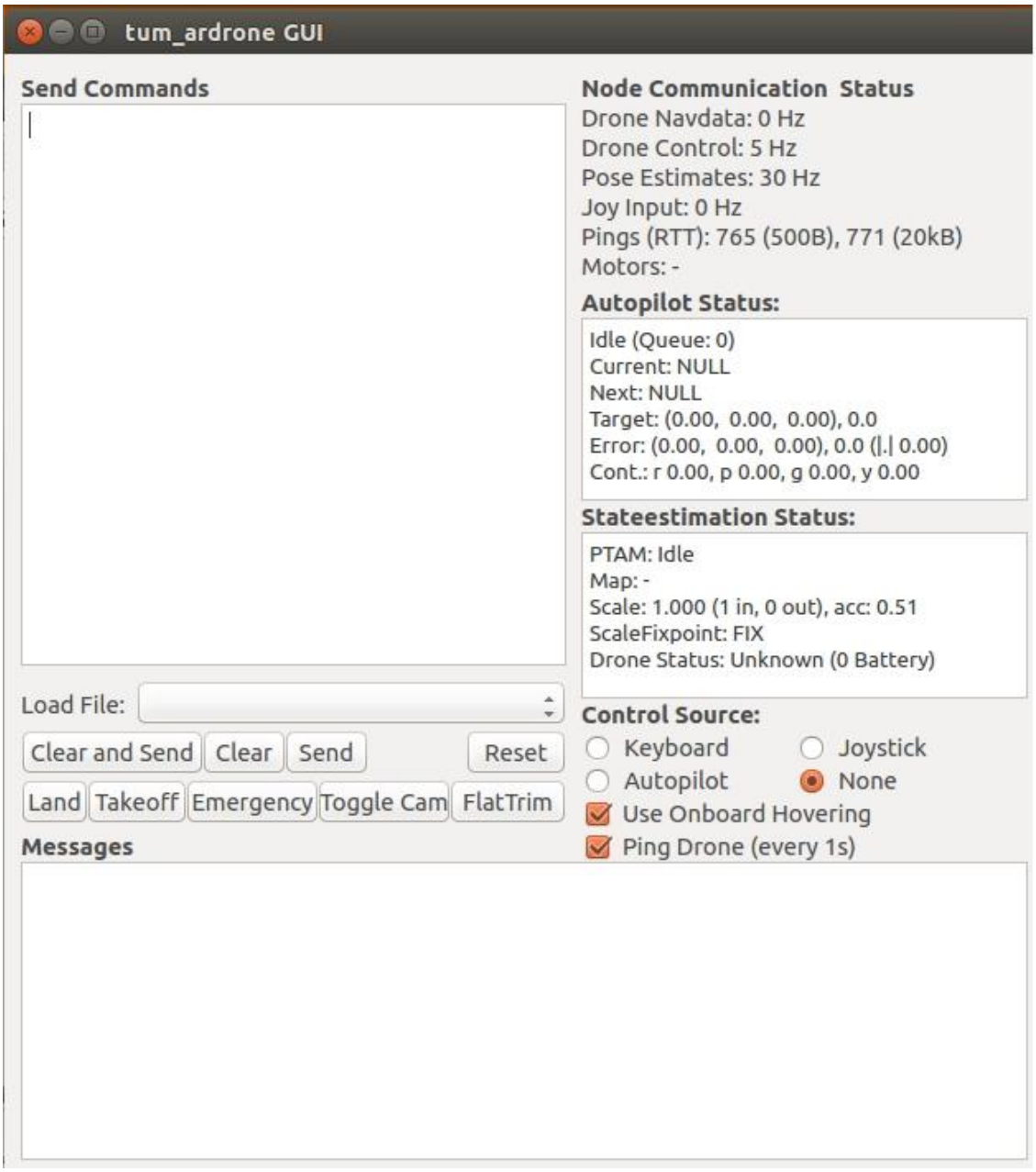


Figure 5 Commans Screen

5.6.2 tum_ardrone Send Commands Screen

When we select the file, which will be sent to the AR Drone, with load file button this screen will appear. In the Send Commands box, you can see the content of the file we selected, which are the commands for the AR Drone. You can clear the Send Commands box by pressing the clear button, and send the commands to the AR Drone by pressing the send button. You can do the both with the Clear and Send button. You can also send commands from the terminal.

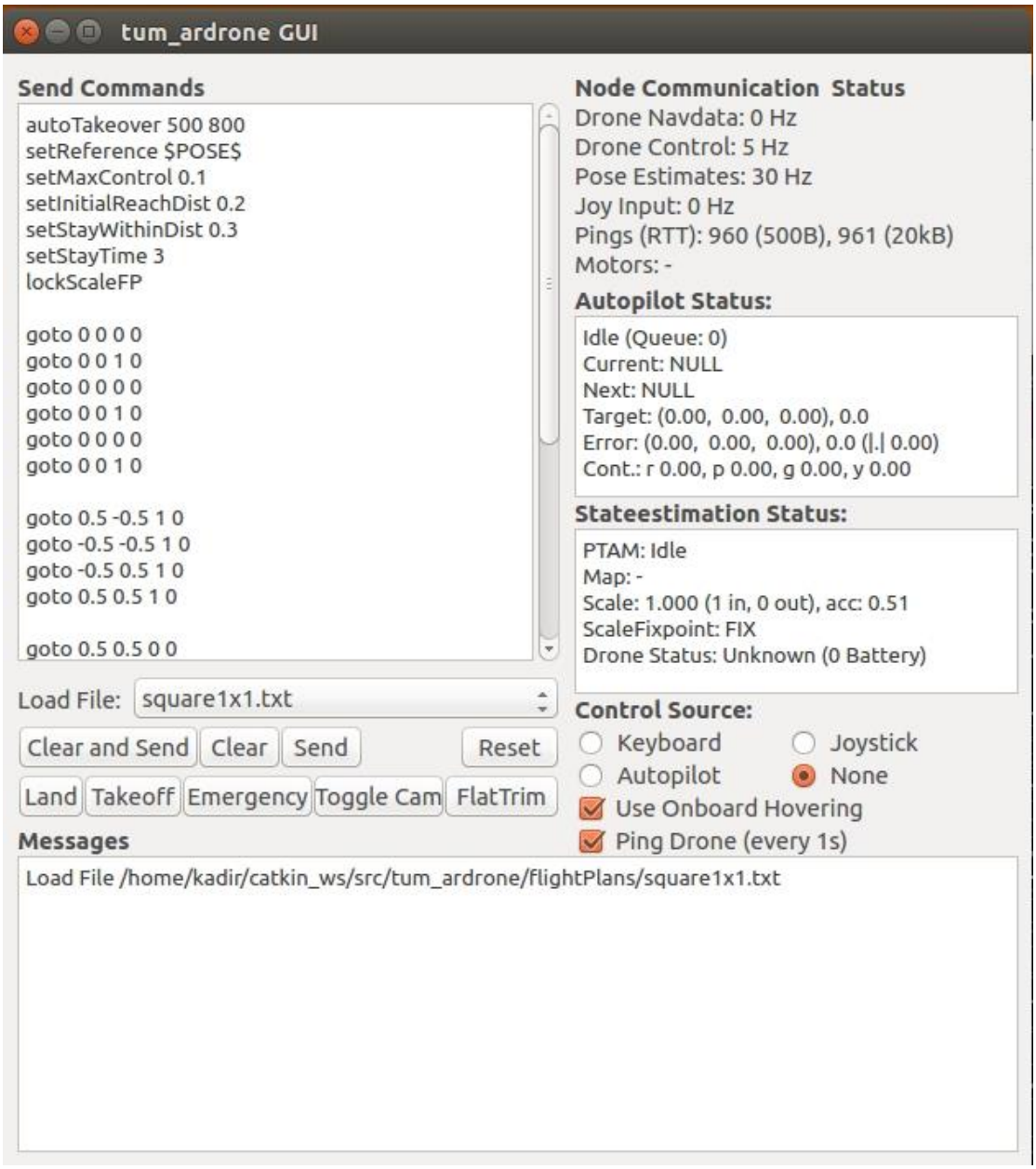


Figure 6 Sending Commands Screen

5.6.3 tum_ardrone Drone Map Screen

This screen will appear with the screen in the 5.5.1 section, when you start the ardrone, this screen shows us the position of the ardrone. When you send commands to the ardrone, the cross sign in the center will move according to the commands.

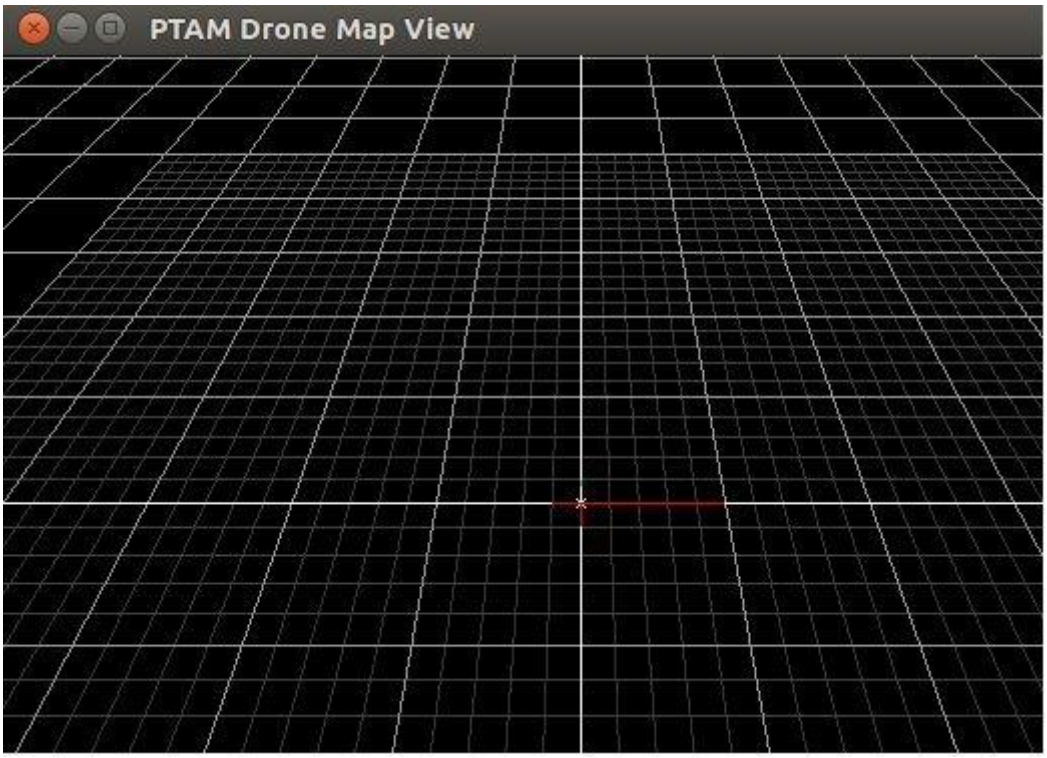


Figure 7 Drone Map Screen

5.6.4 tum_simulator Screen

When you start the tum_simulator, this screen will appear. You can move the ardrone in the screen with keyboard, joystick and commands from the terminal. We will send our commands from the terminal, but we will use the joystick if something will go wrong as a panic button. We will use this to monitor the robots movements from the server.

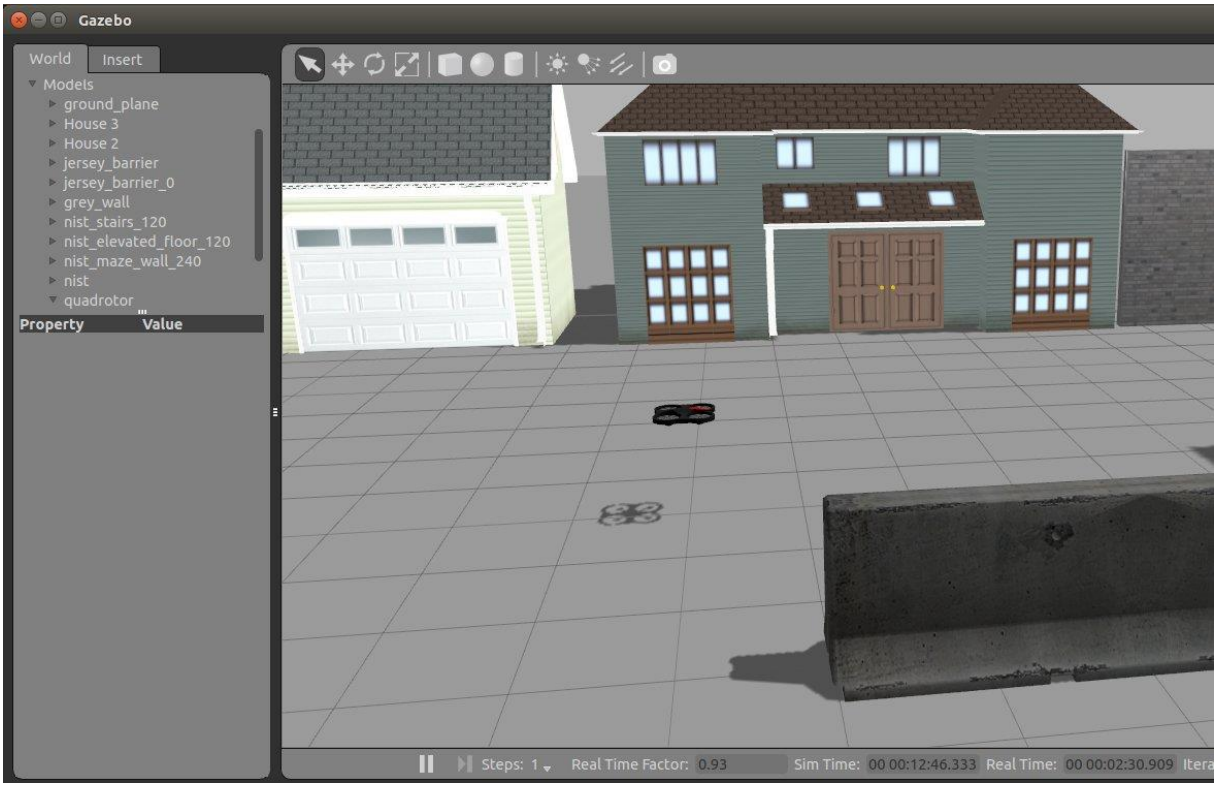


Figure 8 Simulator Screen

5.7 Interaction Viewpoint

In this interaction viewpoint, the way that the system components interact with each other is explained.

System will start with UAV phase, and then it will continue with UGV and server phases. UGV and server phases will happen simultaneously. And according to the scenario user will have selected, system will decide whether these phases will happen in loop or not.

System will be started with the input from the user. After the user selects the scenario and gives the start command, system will be active. First, UAV phase will begin, in this phase UAV will take off , and it will start hovering. It is going to take photos from certain points while hovering. It will send those photos to the server and UGV, for processing and running the algorithms, as it go. After scanning the certain area, it will land on the UGV, then UGV and server phases will start. And according to the scenario user will have selected, the system will decide whether this will happen in loop or not. And it will continue until UAV finishes the scanning the whole area, and UGV reaches the target or exit. In the UGV and server phases, UGV and server will fulfill their duties. UGV will receive the images and process them by running the CV algorithms, then it will run the AI algorithms, and it will start following the calculated path until it reaches the target or exit. Simultaneously server will receive the images and run the simulator, until UGV reaches the target or exit.

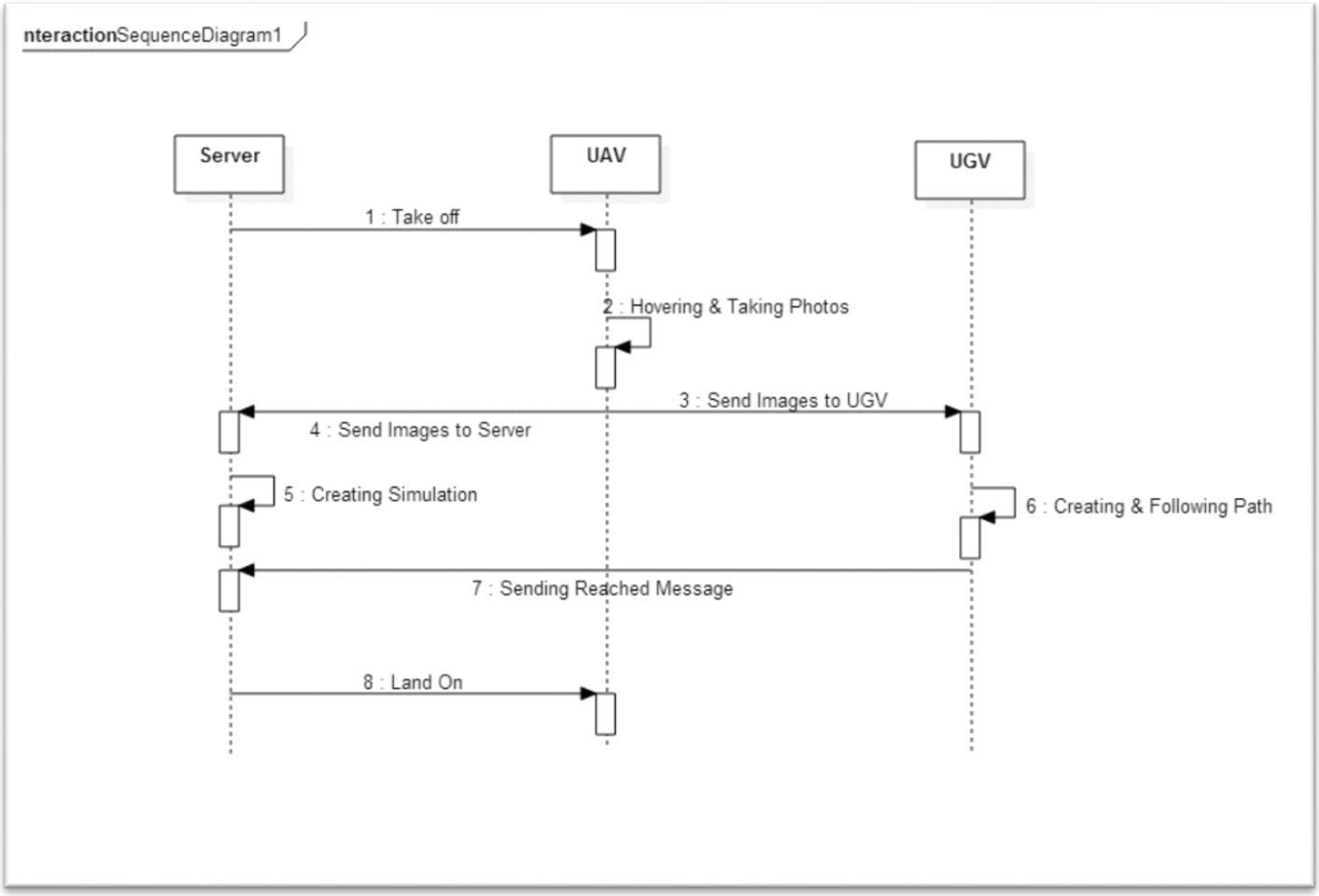


Figure 9 Sequence Diagram

5.8 State Dynamics Viewpoints

This section explains system and user functionalities. There are three state diagrams in our project, and these diagrams are connected to each other, and each one explains a different phase.

Our system will start with UAV phase, and then it will continue with UGV and server phases. UGV and server phases will happen simultaneously. And according to the scenario user will have selected, system will decide whether these phases will happen in loop or not.

System will be started with the input from the user. After the user selects the scenario and gives the start command, system will be active. First, UAV phase will begin, in this phase UAV will take off , and it will start hovering. It is going to take photos from certain points while hovering. It will send those photos to the server and UGV, for processing and running the algorithms, as it go. After scanning the certain area, it will land on the UGV, then UGV and server phases will start. And according to the scenario user will have selected, the system will decide whether this will happen in loop or not. And it will continue until UAV finishes the scanning the whole area, and UGV reaches the target or exit. In the UGV and server phases, UGV and server will fulfill their duties. UGV will receive the images and process them by running the CV algorithms, then it will run the AI algorithms, and it will start following the calculated path until it reaches the target or exit. Simultaneously server will receive the images and run the simulator, until UGV reaches the target or exit.

5.8.1 UAV Phase

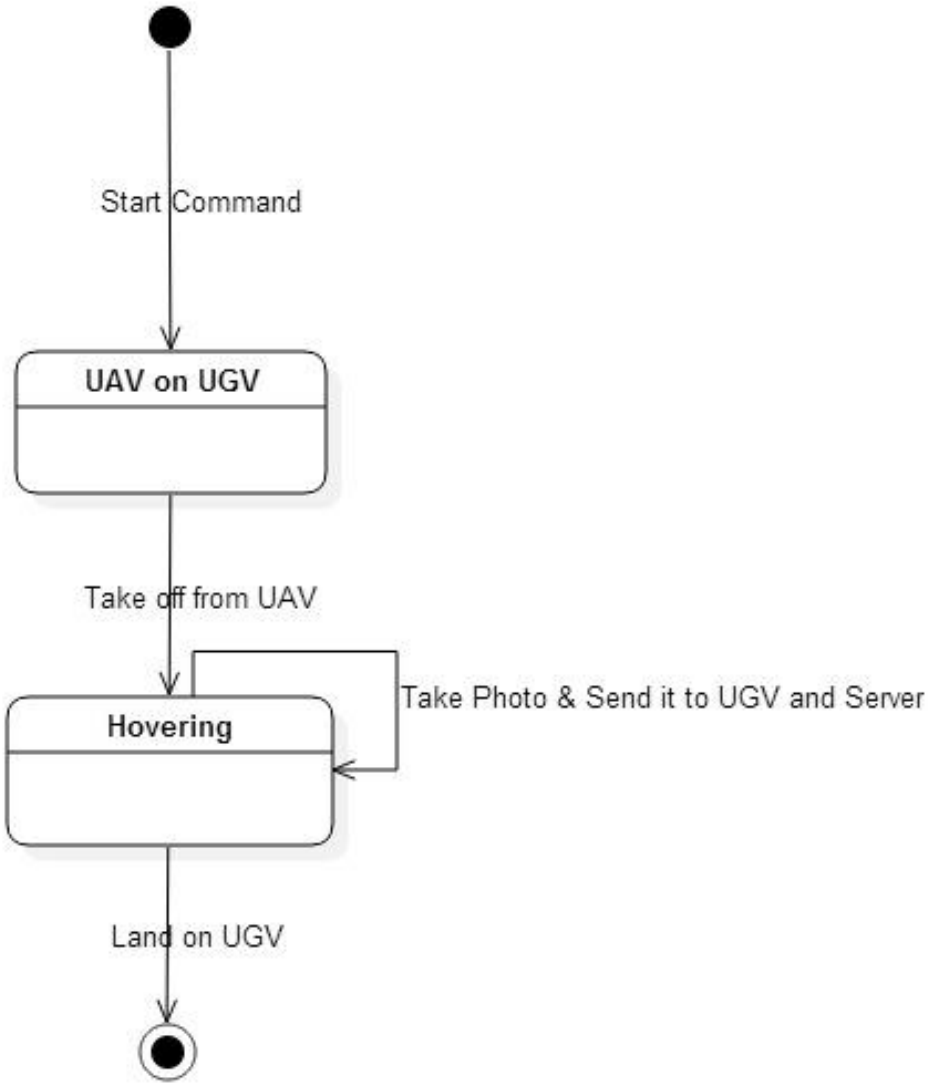


Figure 10 UAV State Diagram

The Above diagram explains the phase where UAV is active.

5.8.2 UGV Phase

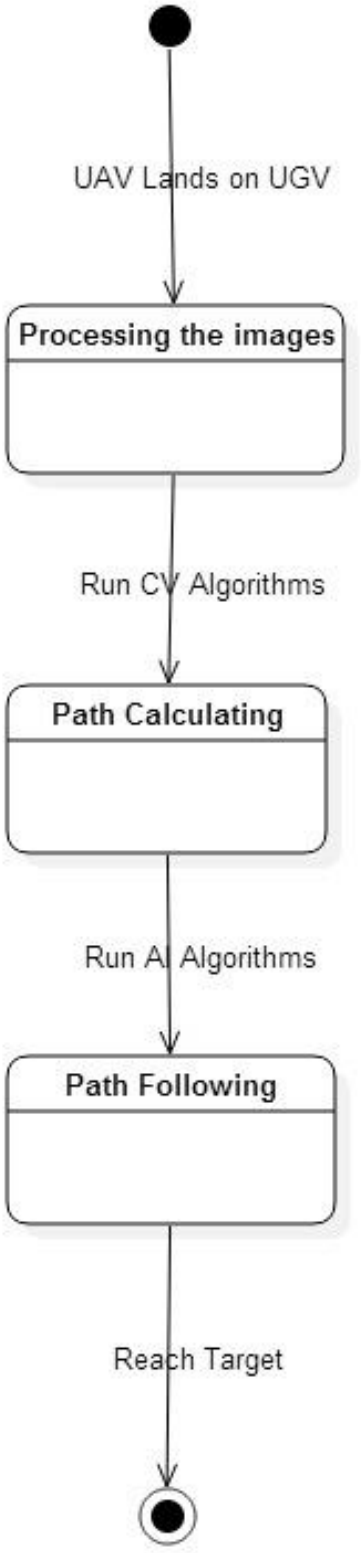


Figure 11 UGV State Diagram

The Above diagram explains the phase where UGV is active.

5.8.3 Server Phase

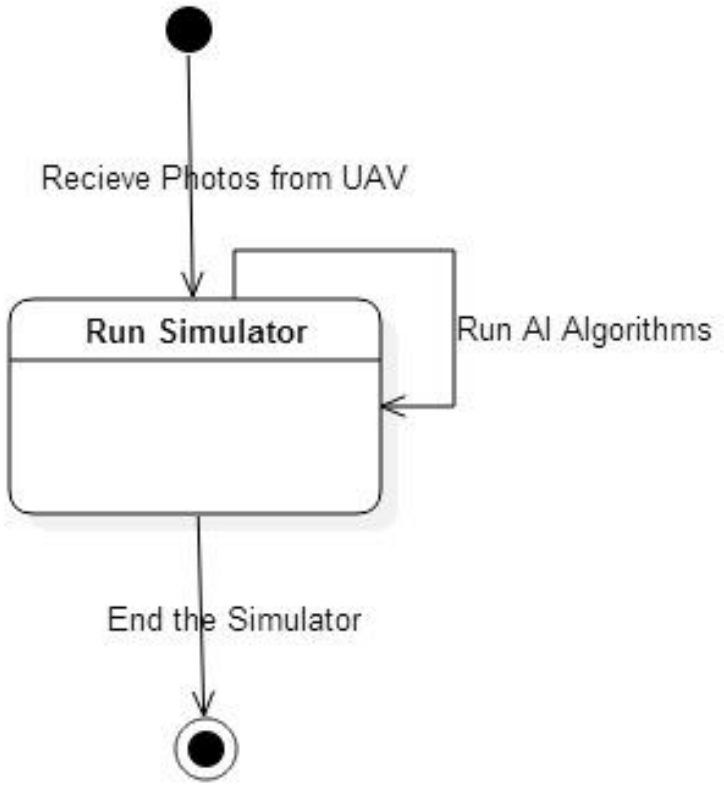


Figure 12 Server State Diagram

The Above diagram explains the phase where Server is active.

6 ESTIMATED SCHEDULE

Table shows after groups and projects took shape and project documents prepared.

Table start week date is 3.11.2014.

Estimation Weeks	Deadline	Task Explanation
Weeks 1,2,3	24.11.2014	Setup ROS & Gazebo on personal computer. Creating a basic Quadrocopter simulation.
Weeks 4,5,6	15.12.2014	Making Test Flights with actual Quadrocopter
Weeks 7,8	29.12.2014	Creating a basic Autopilot for Quadrocopter simulation.
Weeks 9,10,11,12	26.01.2015	Creating and deploying a stabilization algorithm for Quadrocopter
Weeks 13,14,15	16.02.2015	Creating a more advanced Autopilot for Quadrocopter simulation
Weeks 16,17,18,19	16.03.2015	Creating final Autopilot which is capable of following every route
Week 20	23.03.2015	Working on manual control of UAV with Joystick
Weeks 21,22,23,24	13.04.2015	Work on Scerio 1
Weeks 25,26,27,28	11.05.2015	Work on Scerio 2
Weeks 29,30,31	05.06.2015	Work on Scerio 3

7 CONCLUSION

This Software Design Description Documentation contains the system architecture and implementation details of the Raven Nest project. In this document, basics of data design, modules and design viewpoints of the system are described respectively. The software application and packages that will be used while designing and developing the system are also identified. The design topics are specified along with the related design viewpoints. The chart(Part 6) that covers the Raven Nest project stages and procedures of this semester and next semester which is provided in order to indicate the important milestones and schema of the project.