Rescue on wheels project documentation

Team 2

Damian de Hoog 500780277
Yoshio Schermer 500760587
Mustafa Yücesan 500769574
Mohamed El Hadiyen 500777214

November 20, 2018

Contents

1 Introduction

This document contains all the information regarding "Metabot" our rescue on wheels project. This document will describe the design phase, working phase and post-launch phase consisting of:

- An analysis of the requirements for the project from the perspective of an user.
- A design model which explains in great detail how the system is built.
- Documentation containing important code segments with comments and explanations as well guidance for installation, operation and maintenance.

Metabot is a Raspberry Pi powered mobile robot designed to assist in rescue operations. The robot will be able to navigate difficult to traverse environments and explore areas as a sort of reconnaissance unit.
Metabot is equipped with a camera which will broadcast to a mobile device on which the user can control the robot as well as see said camera feed. The camera has a facial recognition functionality to assist in spotting survivors.
Whilst navigating the operating environment Metabot will map the area and ping locations on the map when survivors are found.

Metabot will be able to explore and create a map of the area with the locations of survivors. This way, having seen the environment and knowing the locations of the survivors, the rescue team will be able to conduct a swift and efficient rescue operation.

The following chapters will describe the robot in more details as well as it's design. We will begin with the analysis of the requirements for the Metabot.

2 Analysis

The analysis consists of the following parts:

- Epic
- User stories
- Detailed requirements
- Use-case diagram
- Domain model

The formulation of the requirements was performed based on user stories. These user stories were created based on an epic. An agile epic is a body of work that can be broken down into specific tasks (called "stories," or "user stories") based on the needs/requests of customers or end users[1].

After having created said requirements we will formulate the Use-case diagram(s) and the Domain model(s). These will visualize the user interaction with our system and our Metabot.

2.0.0.1 Terms and definitions

Below you'll find the terms and definitions used throughout this document.

Terms	**Definitions**
Rover	Remotely-controllable RC-car with sensors attached to it.
App	The mobile app through which the rover can be remotely-controlled.
Rover operator	An individual who operates the rover.
Web interface	The web interface through which the rover can be remotely-controlled.
Livestream	A livestream of the camera feed that is attached to the rover.

Table 1: Terms and definitions for this document.

[1] https://www.atlassian.com/agile/project-management/epics

2.1 Epic

A building has collapsed trapping the people inside. Some managed to get out in time but others weren't so lucky. Upon arrival at the scene our actor assesses the situation. The building is unstable and the trapped survivors need to be rescued as quickly as possible.

Normally a rescue crew would be assembled and they would slowly make their way through the rubble to search for people. This is a dangerous and time-consuming task. The crew has no idea whether or not there is a way to get to the survivors. Removing rubble could make the building collapse even further. These people need to be found and removed from the ruins before this happens.

Luckily our actor has just the tool for this job. The Metabot, a small remote controlled robot that can do the scouting for our actor. Our actor takes out his phone and boots up the app. Here, our actor can control the robot.

Our actor sends the robot into the ruins, slowly making it's way deeper and deeper into the building. Whilst moving, the robot sends out signals to its surroundings to map the area and to avoid collision with objects. Meanwhile, our actor can see what the robot is seeing through the live camera feed. Our actor uses this feed to try and find survivors while also looking around for possible routes to take with the rescue crew.

Once our actor has found a survivor, the robot's facial recognition software will help our actor detect the survivor. After this detection, the robot will ping the location of the survivor on the map so that our actor knows where in relation to the rest of the building the survivors are. After pinging the location, the robot will give a visual light signal to the survivors to let them know they have been found and will soon be rescued.

After mapping the area and locating the survivors our actor can create a rescue plan and execute it. Having performed reconnaissance safely and quickly with the Metabot, our rescue crew can now swiftly save the survivors.

2.2 User stories

From this epic story we could formulate user stories, this resulted in the following table:

M: As operator I want to be able to recognize a survivor's face, so I can get information about this survivor.	**M: As operator I want to visually explore the environment from a distance, so I can be better prepared for the rescue operation.**	**S: As operator I want to know the locations of the survivors in relation to the environment, so that I know where I can find the survivors.**	**S: As a survivor I want to know if I have been found, so I can be rescued.**
As operator I want a mobile robot to recognize a survivor's face, so I can get information about this survivor	As operator I want the mobile robot to show me the environment on a screen at a distance so that I can plan the rescue operation accordingly.	As operator I want a mobile robot to tell a mobile app where the survivor is in relation to the environment, so that I know where I can find the survivors.	As survivor I want the mobile robot to give me a sign, so I know I have been found.
As operator I want a mobile app to get the information that corresponds to the survivor's face, so I know who the survivor is.	As operator I want the mobile robot to stream the camera feed to a mobile device so I can get real time information from the environment.	As operator I want a mobile app to display where the survivors are in relation to the environment, so that I know where I can find the survivors.	
	As operator I want to control the mobile robot from a distance to plan the rescue operation accordingly.		

Table 2: User stories formed in accordance with the epic story.

2.3 Use-cases

Use Case ID:	1
Use Case Name:	Connect with rover
Primary Actor:	Rover operator
Secondary Actor:	Rover
Pre-conditions:	1. Rover operator has mobile app or address of web interface. 2. Rover is nearby.
Success Guarantee:	Rover operator successfully connected the mobile app/web interface to the rover.
Main Success Scenario:	1. Rover operator opens app or goes to the web interface. 2. Rover operator searches for rover to connect to within the app. 3. Rover operator selects rover to connect to. 4. Rover operator confirms to connect to the selected rover. 5. Rover operator successfully connected the app/web interface to the rover.
Exceptions	• Rover operator already has established a connection between the app/web interface and the rover. • Rover operator cannot connect app/web interface to the rover, because the rover is already connected with the same app/web interface from another rover operator.
Special Requirements	-

Table 3: Use-case 1

Use Case ID:	2
Use Case Name:	Look for survivors on flat surfaces
Primary Actor:	Rover operator
Secondary Actor:	Rover
Pre-conditions:	1. Rover operator has established a connection between the app /web interface and the rover. 2. Rover operator sees what the rover sees. 3. Rover operator has controller.
Success Guarantee:	Rover operator uses the rover to look for survivors on flat surfaces, that is, drives around.
Main Success Scenario:	1. Rover operator uses controller to drive either forward, backward, left, or right. 2. Rover operator uses the live stream to look for any survivors.
Exceptions	-
Special Requirements	-

Table 4: Use-case 2

Use Case ID:	3
Use Case Name:	Know who a survivor is and where they are
Primary Actor:	Rover operator
Secondary Actor:	Rover
Pre-conditions:	1. Rover operator has established a connection between the app/web interface and the rover. 2. Rover operator sees what the rover sees. 3. Rover operator has controller.
Success Guarantee:	Rover operator drives around with the rover. Through the livestream the rover operator can see who the survivor is and where they are to be seen.
Main Success Scenario:	1. Rover operator uses controller to drive either forward, backward, left, or right. 2. Rover operator uses the live stream to look for any survivors. 3. A survivorâĂŹs face is visible through the livestream. 4. Through the livestream the rover operator can tell who the survivor is and where the face of the survivor is.
Exceptions	-
Special Requirements	-

Table 5: Use-case 3

Use Case ID:	4
Use Case Name:	Tracking the location of the rover
Primary Actor:	Rover operator
Secondary Actor:	Rover
Pre-conditions:	1. Rover operator has established a connection between the web interface and the rover. 2. Rover operator has controller.
Success Guarantee:	Rover operator tracks the location of the rover through a map on the web interface
Main Success Scenario:	1. Rover operator uses controller to drive either forward, backward, left, or right. 2. Rover operator sees location of the rover changing in the same direction as he/she is moving it.
Exceptions	-
Special Requirements	When the rover is unable to move in a certain direction because of an obstacle then this should be reflected in the map as not having moved.

Table 6: Use-case 4

Use Case ID:	5
Use Case Name:	Backtracking the rover
Primary Actor:	Rover operator
Secondary Actor:	Rover
Pre-conditions:	1. Rover operator has established a connection between the web interface and the rover. 2. Rover operator has controller.
Success Guarantee:	Rover returns to initial position in the opposite way as it has moved to its current position.
Main Success Scenario:	1. Rover operator uses controller to drive either forward, backward, left, or right. 2. Rover operator turns backtracking on. 3. Rover goes back to its initial position.
Exceptions	-
Special Requirements	Moments where the rover stopped are ignored. So, the rover drives back directly without stopping.

Table 7: Use-case 5