RescueMe – Rescue Robot

Md Limon Apu, Arfat Kamal, Abdullah Al Forkan, Tauseef Ahmed

Hochschule Hamm-Lippstadt

Abstract

Every one of the new advancements and all the new examination and investigation fields of the present world, mishaps, and crises additionally emerge. That is the reason new rescue methods should be applied, arrive at the debacle zones in the briefest conceivable time, yet in addition, have the option to save however many lives and frameworks as could be expected under the circumstances. A Rescue robot is a choice to assist with safeguarding groups to get to the debacle zones in a protected manner. In this undertaking, we fostered a model for a Rescue robot. We began with the displaying of the framework, understanding the necessities, and the destinations to foster this robot. We broke down the Use cases, and the framework for this issue. Second, we moved to the prototyping, where we planned the actual robot, and we got done with the programming in a Rescue situation. During this interaction, we dealt with various issues, for example, the confronting changes in the necessities, or changes in the plans to get the ideal measurements. Likewise, we discovered new apparatuses to foster a successful calculation that will assist the robot with satisfying its main goal. In the end, the main goal of RescueMe robot is to save people's life and rescue objects from a rescue situation.

1 INTRODUCTION (Md Limon Apu)

A robot cannot harm a person, nor can a person be allowed to harm because of its inaction. This is a real case where our rescue robot hopes to rescue victims of accidents and emergencies. The project is divided into different parts but indispensable parts; system technology, prototyping and programming. The design of the system is the field of irrigation where everything started. From development requirement diagrams to use case diagrams, everything is to make robots better and smarter, and have a first-class understanding of various tasks. The development of the system provides a clear understanding and realization of the robot. Once you understand the system requirements and have a mind map for developing the robot, the prototype design will take effect. With the prototype, we can first design the robot based on the sketch, and then use the CAD software SolidWorks. The construction is carried out in accordance with technical specifications and standards. The final part of the prototype design is to determine the various electrical/electronic components that will be used to develop the robot and confirm its suitability. Robots are as smart as software. This is where project planning comes into play. We can write code to detect robot movement. Although the code we write has no direct impact on our robots, we have used robotic stimulation to write code that is known to take the best route and save energy. Share tripartite contributions to systems engineering, prototyping and software programming. We also believe that our robot can add many modifications and extensions.

2 Diagrams (Md Limon Apu)

2.1 Requirement Diagram

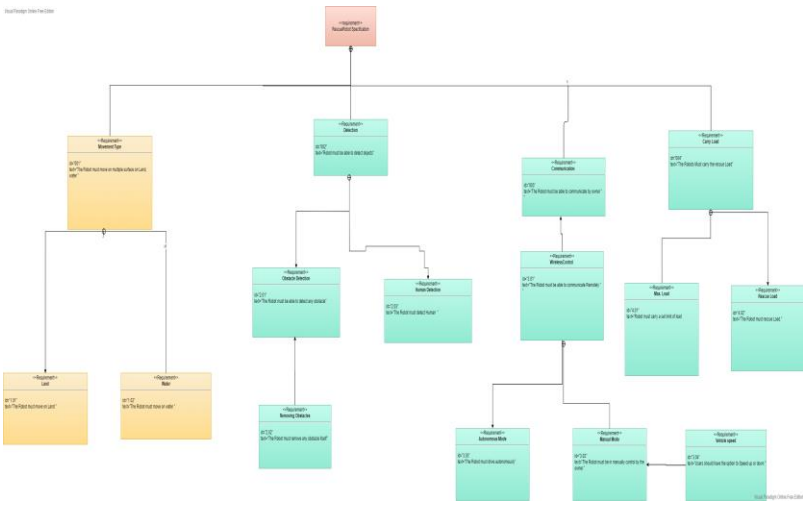


Fig. 1. Requirement Diagram

A requirement diagram is a static structure diagram that shows the relationship between the requirement structure (requirements), the model elements that satisfy them, and the test cases that test them. Non-functional requirements in the model so that they can be traced back to other model elements that satisfy them and the test cases that verify them. As can be seen from the requirements below, we can see how the functional and non-functional requirements in the model meet each other in terms of communication. Measure important parameters on the vehicle until the rescue robot detects the object.

2.2 Activity Diagram

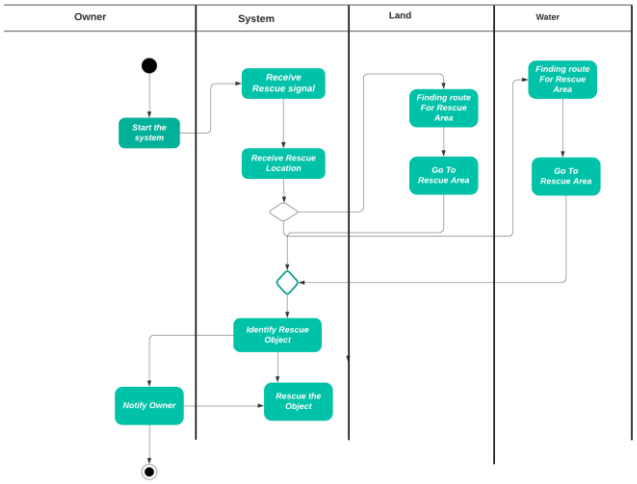


Fig. 2. Activity Diagram

The activity diagram shows the actions that will be performed in the scene of the rescue robot system. In the action diagram, the user activates the rescue robot system and immediately goes to the rescue location to determine the best route on land or in water. The rescue robot system analyses the position and initializes the motion system to drive to the rescue position. Rescue Determine how to rescue or remove obstacles. After detecting the object, the system loads the object for rescue, whether it is a human victim or an object. The same process occurs until the end of the search.

2.3 Use Case Diagram

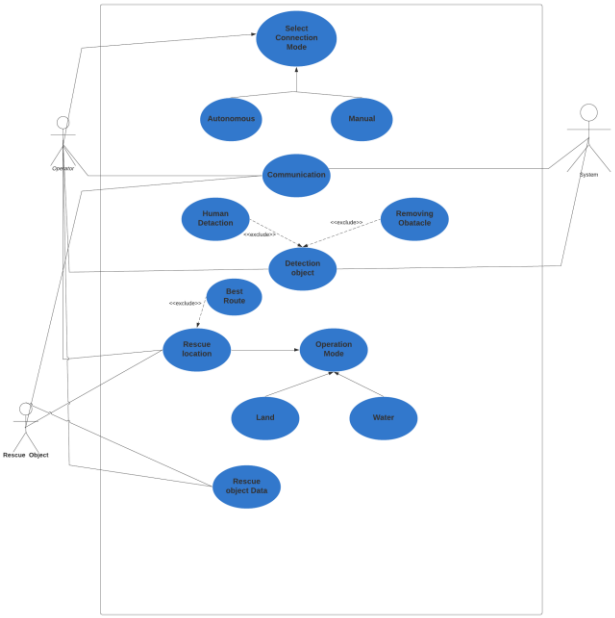


Fig. 3. Use case Diagram

The following use case diagram describes the robot rescue system and the victim detection system. The robot rescue system considers use cases and two participants. The relevant operator interacts with the system through selected control modes, including the use of automatic, manual or hybrid modes. Also, the Use case describe for communication commands and important measures This use case also explains the commands and calls the corresponding use cases to perform functions.

2.4 Sequence Diagram

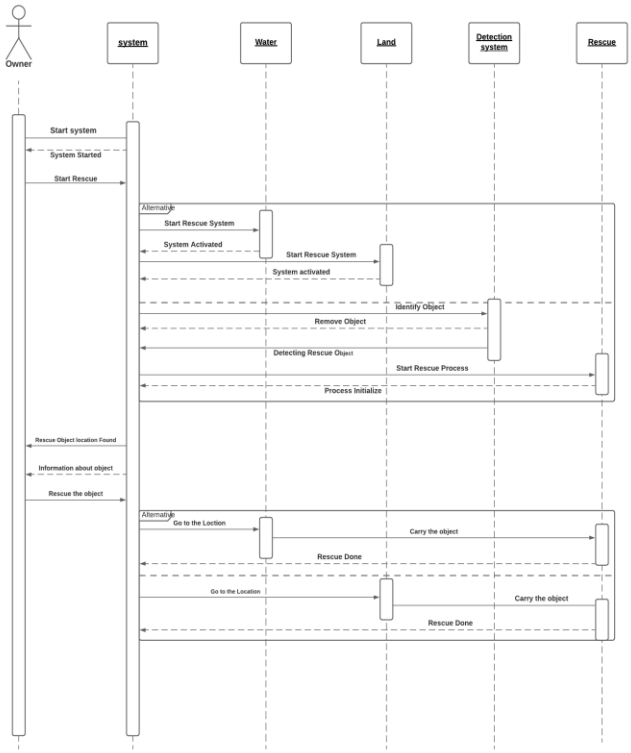


Fig. 4. Sequence diagram

Sequence diagrams are used to describe the interaction between objects and classes in the robot rescue system. It can also be used to model the relationship between time-ordered block structures. The following figure shows the Sequence chart of the operator and rescue robot. When the operator starts the rescue robot system, he will send a command to the system to initialize the drive system for road/land and water modes. After the connection is established, the system and the rescue robot can communicate and exchange data objects. And the classes in the system

3. PROTOTYPING (MD LIMON APU)

There is a design behind every great product. Excellent design defines an excellent product in terms of appearance and function. It takes a lot of creative thinking and hard work to develop a Rescue robot. The way robot is built is to stable and rescue the rescue object. The shape and structure give the Robot manoeuvrability and flexibility when moving. The same thing makes them powerful and cruel. This is what our robot needs. Since our robot must work under adverse conditions, this design helps us to work there smoothly. Our design allows our robot not only to operate on land, but also on water.

3.1 Original Concept

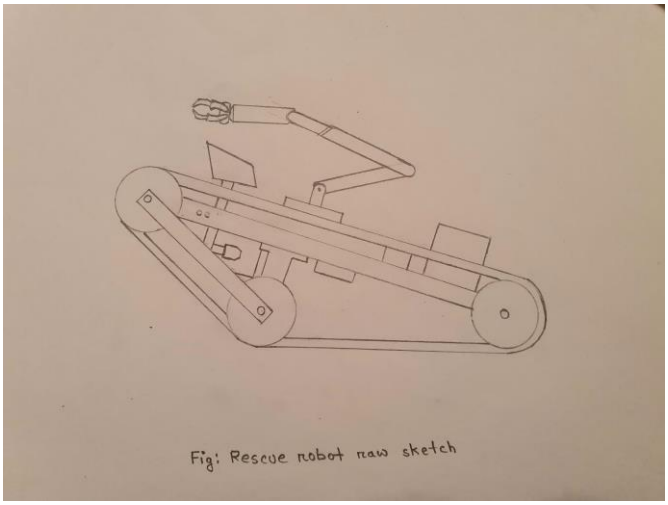


Fig. 5. Original drawing

The first concept was to have a rolling wheel too operate on land and water. And the front part has sensors and camera for the detection system for the rescue object. And the robotic arm will grab the rescue object with a minimum object width and length. The design was inspired by army design where some tank can go on water and land.

3.1 Robot Arm

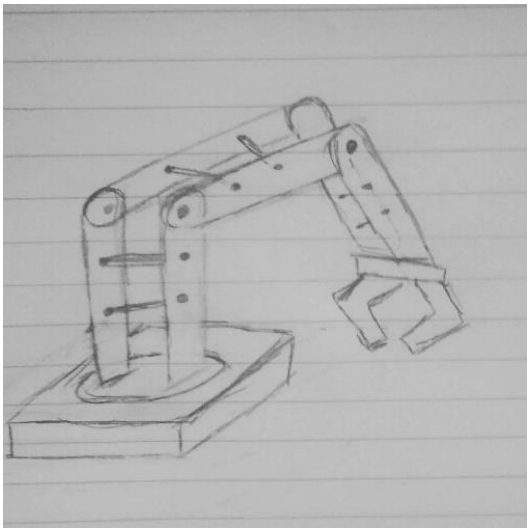


Fig. 6. initial arm drawing

3.2 Changed Robot Arm

Fig. 7. New robotic arm builds on Solidworks

The initial robotic arm drawing come with various challenges to build then we come with a simpler and easier approach for drawing and design for the robotic arm.

3.3 Tools & Materials

We used a variety of tools to preserve the excellent design of our Rescue robot. A list of these tools is given below.
Concept:

- A drawing program used to develop the concept of Rescue robots.
- SolidWorks was used to create our 3D prototype.

Servo Motor-

What is important is that the rotating parts of the robot leg have high precision when moving in all directions, and the ability to react quickly to any change in direction is the reason for choosing a servo motor. And each leg is constructed with three rotatable and flexible support pins; joints allow the robot to overcome obstacles. Each hinge is used by the robot, and the servo motor is connected to the hinge of the robot. The degrees of freedom of these legs are unlimited, and the accuracy is higher.

4. Method

The first task is to let the robot find the target on different maps. The first method we use is to find the width of the map, then get the coordinates of the robot and the target and divide them into two dimensions. This method has many disadvantages. The first is that when the robot has to avoid obstacles, the robot has to encounter obstacles. It can enter an infinite loop, approach the target one step, and then avoid obstacles one step at a time. Go back to the previous position and repeat the process indefinitely. The second method we use is to determine what happens if the robot takes a step in each direction. For the first question, if the next step (up, down, left, right) is not a wall, and the distance calculation algorithm is as follows, to the left. Or the right side of the robot receives the next number. If there is a possible movement, this search is performed until the robot finds a new movement, or if the robot finds the target, the same procedure is used for all cells to make the surrounding robot reach a value of 4 At the end, one in each direction. After calculating the number of steps in each direction, the robot will eventually choose the one with fewer steps. In this example, the robot can move down or to the right. Since the algorithm is suitable for obstacles, we tested it with a map containing obstacles between the robot and the target. The position of the obstacle requires the same number of steps to reach the target to show that the algorithm can handle it.

5. Programming (MD LIMON APU)

In the programming and simulation stage, the robot begins to develop its own world view, using multiple symbols to represent each object. We are required to deal with different environments and program robots to achieve their goals. We explained how we complete each required task so that we finally have a robot that meets all requirements. We developed C code functions using Code blocks tools.

Area Scanning-

```
int goingup, goingdown, goingleft, goingright;       // Area scanning
goingup = (world[(robot_y - 1) * width + robot_x] == '#' ? 0 : 1 );
goingdown = (world[(robot_y + 1) * width + robot_x] == '#'? 0 : 1);
goingleft = (world[robot_y * width + robot_x - 1] == '#' ? 0 : 1);
goingright = (world[robot_y * width + robot_x + 1] == '#' ? 0 : 1);

int moves[199];
 int whenup = 199;
  int whendown = 199;
```

This is where we declare the integer variable and and how it will operate going different direction.

Going Right function-

```
if(goingright == 1)  // going right
  {
     int countmoves;
      int allmoves = 1;


     for (int i = 0; i <= 199; i++)
     {
        moves[i] = 0;
     }
     moves[robot_y * width + robot_x + 1] = 1;
     countmoves = 1;
     int targetFound = 0;
     while((countmoves != 0) && (targetFound == 0))
     {
        countmoves = 0;
        for(int i = 0; i <= 199; i++)
        {
           int x = 0;
            int y = 0;
           if(moves[i] == allmoves)
           {
              y = i / width;
              x = i % width;
              if((world[(y - 1) * width + x] == 'T') || (world[(y + 1) *
width + x] == 'T') ||
                 (world[y * width + x + 1] == 'T') || (world[y * width + x
- 1] == 'T'))
              {
                 targetFound = 1;
                 break;
              }

              if((world[(y - 1) * width + x] != '#') && (moves[(y - 1) *
width + x] == 0))
              {
                 moves[(y - 1) * width + x] = allmoves + 1;
                 countmoves++;
              }
              if((world[(y + 1) * width + x] != '#') && (moves[(y + 1) *
width + x] == 0))
              {
                 moves[(y + 1) * width + x] = allmoves + 1;
                 countmoves++;
              }
              if((world[y * width + x + 1] != '#') && (moves[y * width +
x + 1] == 0))
              {
                 moves[y * width + x + 1] = allmoves + 1;
                 countmoves++;
              }
              if((world[y * width + x - 1] != '#') && (moves[y * width +
x - 1] == 0))
              {
                 moves[y * width + x - 1] = allmoves + 1;
                 countmoves++;
              }
           }
        }
        if (countmoves != 0)
          {
           allmoves++;
        }
     }
```

The goingRight function is similar to other three function like (Goingup, Goingdown,Goingleft).

Best Route-

```
    int bestRoute = 5;
     if((whenup <= whendown) && (whenup <= whenleft) && (whenup <= whenright))
         bestRoute = north;
     else if((whendown <= whenleft) && (whendown <= whenright))
         bestRoute = south;
     else if(whenleft <= whenright)
         bestRoute = east;
     else
         bestRoute = west;

     return bestRoute;
}
```

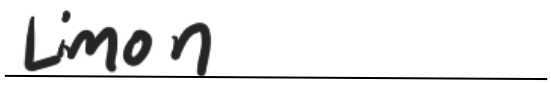
The function determines the best route that the robot can take to rescue the object and avoid the walls.

7 CONCLUSIONS (MD LIMON APU)

After going through the project phases, system modelling, prototyping and simulation, we carefully studied all the key steps that need to be taken when dealing with large projects. Always strive to meet all expectations of the course. This project be the starting point for the development of a full-featured rescue robot, and it is also a major advancement in the way the world's rescue team works.

8 AFFIDAVIT - MD LIMON APU

I hereby confirm that I have written this paper independently and have not used any sources or aids other than those indicated. All statements taken from other sources in wording or sense are clearly marked. Furthermore, I assure that this paper has not been part of a course or examination in the same or a similar version.



Md Limon Apu
Soest, 16.07.2021

References

[1] https://blogs.solidworks.com/tech/2015/12/whats-new-solidworks-2016-mate-controller-feature.html

[2] Wongwirat, O., Paelaong, S., and Homchoo, S. (2009, December). A prototype development of ET rescue robot by using a UML. In 2009 7th International Conference on Information, Communications and Signal Processing.

[3] Boyes, W. (2013). Understanding how ultrasonic continuous level measurement works. Understanding How Ultrasonic Continuous Level Measurement Works. https://www.controlglobal.com/articles/2013/automationtechnology-ultrasonic-continuous-measurement/Power Sources for Small Robots - cs.cmu.edu.