Search and Rescue UAVs and G-Robot

Ahmed Alharthi
Control System Engineering
King Fahd University of
Petroleum & Minerals
Dhahran, Saudi Arabia
g200814040@kfupm.edu.sa

Shabib Aldossary
Control System Engineering
King Fahd University of
Petroleum & Minerals
Dhahran, Saudi Arabia
g201545290@kfupm.edu.sa

Abstract —The desert is covering a massive area of Saudi Arabia's land and over the years, several cases reported of missing people in the desert. Unfortunately, the rescue teams could not rescue all the victims, which result in some fatalities cases [1]. One of the major reasons for death cases is the duration taking a long time for the rescue teams to reach the victim, especially in rough desert terrain. The main objective behind the proposed project is to implement an idea that focuses on saving lives, where every day, people go missing in different difficult terrains, especially in Saudi Arabia. Due to that, we developed a simulation system where the major function is to find the lost people in the desert in a shorter time. The system is contained two Unmanned Ariel Vehicles and one Unmanned Ground Vehicle (UAVs & UGV), where both drones (UAVs) will search in a given area using a camera, to detect any missing individuals in that area. Afterward, if any missing individual is located, a ground robot (Husky) will be dispatched to the given location where it carries first aid kits and food rations which will help them to stay alive until the rescue arrives.

Keywords—rescue, search, UAV, UGV, drone, husky, target

I. INTRODUCTION

The Arabian Peninsula's most prominent feature is the desert, of which Saudi Arabia is the largest country. Over the years, many people were lost across Saudi Arabia's desert. Currently, there are several rescue teams in the kingdom, and the teams have specially equipped cars and camera planes for searching and rescuing the missing people in the desert. However, some of the missing people were found dead and there wasn't much time to save them. The proposed idea is implementing a system that could help to save human lives by simulation two Unmanned Ariel Vehicles (UAVs) and one Unmanned Ground Vehicle (UGV) through ROS (Gazebo).

Given a scenario of a missing person being reported to be lost in a desert area of 50 by 50 km in Tanajib, most definitely his car has malfunctioned, and he is stuck in the middle of nowhere.
The UAVs (drones) will be given a path to cover the reported area and scan it for a car or a human, when the UAV detects the lost individual, his coordinates will be shared with the rescue team and a ground robot (husky) will be dispatched to the given location where it carries first aid kits and food rations.
For the simulation using ROS packages and gazebo, where the packages are hector_quadrotor_notic [2] for the drones and husky [3] for the ground robot.
When the two drones are notified and start launching, they will fly in a spiral movement. Then, if one of them detects the target through the onboard camera, they will send the target's coordinates to the ground robot (husky).

II. BACKGROUND

A. Robot Operating System

The Robot operating system (ROS) is an open-source framework, and it is something between middleware and a framework built for robotic applications. The goal of ROS is to provide standards for robotic software that developers can use in any robot. ROS is a strong software base to directly program high-level functionality by utilizing the packages that already exist in ROS or using other packages, which should be compatible with the ROS version [4].

B. Gazebo

The gazebo is a free robotic simulation environment, which creates a virtual world and loads a simulated version of robotics into it. The gazebo is integrated with ROS and simulates sensors that can detect the environment and publish the data to the same ROS topic that real sensors would do [5].

III. LITERATURE SURVEY

A. Article Summary Table Reference

Tsunami post-disaster robot simulation and implementation of aerial path planning to explore unknown environment | Dr. Eny Kusumawati - Academia.edu [6]

B. Introduction

A tsunami is one of the catastrophic dangers to many coastal communities in Indonesia. The authorities are still heavily reliant on human labor in the search and rescue effort, while the lives of the rescuers are still at stake and critical to avoid more accidents. Given these circumstances, disaster robot aid is desperately needed. The idea is a development a UAV with the features of 2D local aerial path planning with obstacle avoidance.

C. Main body

The local path planning and obstacle avoidance approach are based on customized behavior-based control and the motion planning algorithm Vector Field Histogram and VFH+. Customized behavior-based control is used to develop the robot's skills for avoiding obstacles in an unknown environment. While avoiding collision behavior, the VFH+ algorithm is used. The quadcopter robot avoids obstacles en route to the specified point based on algorithm calculations

and modified cost function parameters. The distance traveled by the prototype quadcopter robot while performing the maneuver is determined by the cost parameters defined on the program.

D. Main result

RViz 3D visualization tool for Robot Operating System (ROS) Melodic Morenia on Ubuntu 18.04 is utilized for simulation. The simulation environment involves all the components used for the simulation, such as the robot model, sensors, and static objects. The information from the laser distance sensors is presented as red dots around the quadcopter. The path waypoint and path actual nodes in RViz can be activated to display the flight route.

E. Conclusion

The team was able to model and prototype 2D local aerial path planning for a quadcopter acting as an aerial disaster robot to explore an uncharted area. The simulation outcome is demonstrated by the quadcopter's flight trajectory when flying to the designated area. In addition, the quadcopters were able to avoid the obstacles. The successful implementation of the VFH+ algorithm leads to enhancing the sensor reading. The greater the safety distance and turning radius variables, the more responsive the quadcopter's mobility while avoiding obstacles.

F. Comment

Telemetry has a restricted baud rate, which could lead to failing the implementation mission of aerial path planning.

IV. SYSTEM DESIGN

The system design discusses the system's use cases and activity diagram.

A. Use Case

As shown in figure (1), the proposed system is identified by listing the actors and using use cases to describe the interaction relationship among the actors.

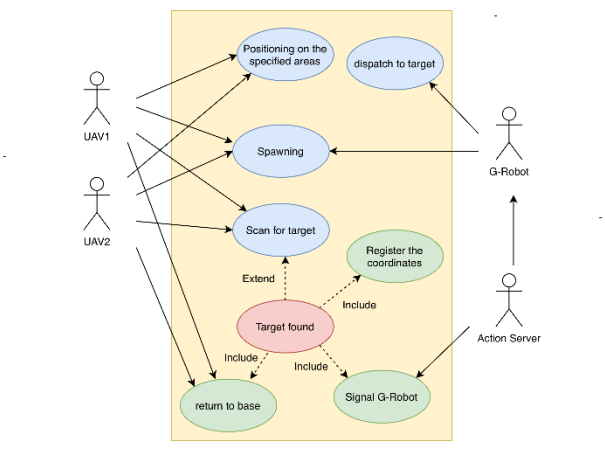

Figure 1: Use-Case Diagram

The first used case is positioning on the specified areas, where UAV1 & UAV2 are the actors. Both drones are to be in position for the target and they will be stimulus through the launch file. The second used case is spawning, and the actors are both drones and the ground robot (husky). Both drones search for the target and the husky will dispatch to the target and they will be stimulus through the launch file. Dispatch to target is the third used case and it uses the husky robot. The husky robot will dispatch to the target once it received the target coordinates, and it will be stimulus by an action server. The fourth used case is scanning the target, which is the stimulus through both drones. The next used case is finding the target with both drones. The target will be found after scanning by drones' cameras, and the stimulus is the detection of cameras. Register the coordinate is the sixth used case and using both drones as actors. Finding the target is the stimulus for registering the coordinates. The seventh used case is to signal the husky robot by the action server. The action server will send the coordinate of the target to the husky robot if the target is found. Return to the base is the last case. Both drones will return to the base after identifying the target.

B. Activity Diagram

The following activity diagram shows the sequence of different possible activities in the system as shown in figure (2).

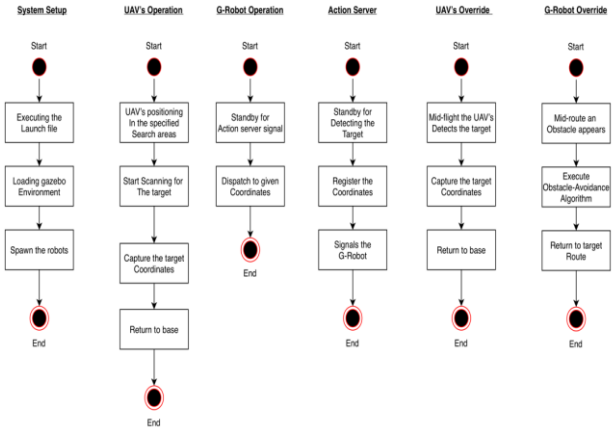

Figure 2: Activity Diagram

The diagram is illustrating the sequence from the start to the end of the system setup, the operation of the drones and the ground robot, the action server, and the override of the drones and ground robot.

Execution of the launch file is the starting point for system setup. Then, loading the gazebo environment and spawning the robot. Operation of UAV is stated by positioning the drones in the specified search area. After that, the drones are scanning the target to capture the target's coordinates. Then both drones will return to the base. The operation of the ground robot is started by receiving the signal from the action server to dispatch to the target's coordinates. standby for detecting the target is the start point of the action server. The action server will register the coordinates and send the signal to the ground robot. UAV's override could be started by mid-flight the drones detect the target. The drones will be scanning the target to capture the target's coordinates and

return to the base. Mid-route could be the starting point for the ground robot's override, and to overcome this obstacle by executing the algorithm of obstacles avoidance.

V. Experimental Setup

The proposed system has been tested on ROS noetic and gazebo simulator. ROS and gazebo are installed in ubuntu 20.04 and the steps to install them are explained in details on the ROS organization website [7].

A. Gazebo environment

We were able to integrate a desert world in the gazebo mimicking Saudi Arabia's deserts.

B. Launch the UAVs

As start UAVs will take off to a height of 30 meters. After that, they will position themself in the specified areas. Moreover, they will start searching for the target. Once the target has been located, the images will capture by onboard cameras. The images will be processed to identify the target's coordinates and signal to the action server as the target is found, and shared with the rescue team.

To calculate the camera to the target distance, the algorithm is applicable to a range of x and y. Where we add an assumption of a known captured range in the middle point of (x,y) as our known distance measured.

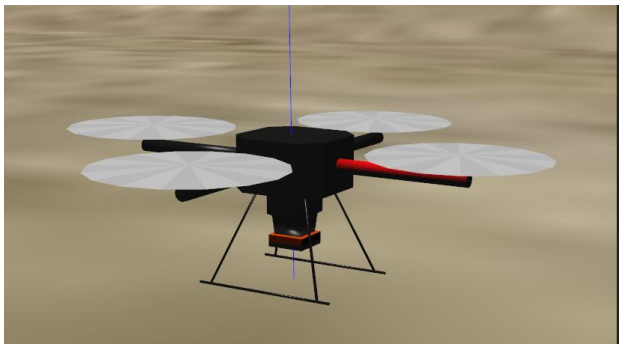

Figure 3 hector quadrotor

C. Launch the UGV

After installing the husky package, we developed a go_to_ goal function that will able the ground robot to go to the desired coordinates. Importing the libraries such as odometry for subscribing the ground robot's pose, importing euler_from_quaternion to get the angular angle of the robot, and importing laser scan to use it in obstacle avoidance algorithm.

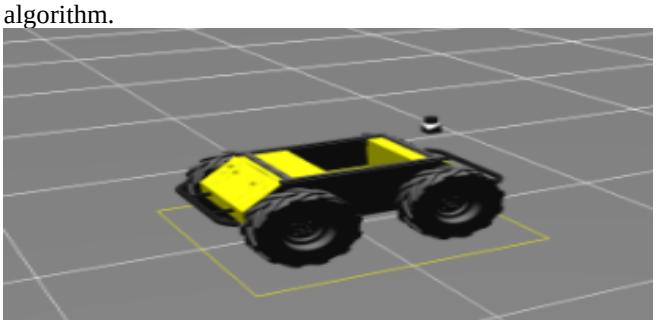

Figure 4 Husky

The ground robot's linear and angular speeds need to be measured for reaching the goal coordinates. To get the linear speed, the distance between the desired and the actual coordinates need to be calculated. The formula for the distance is shown in figure (3) [8].

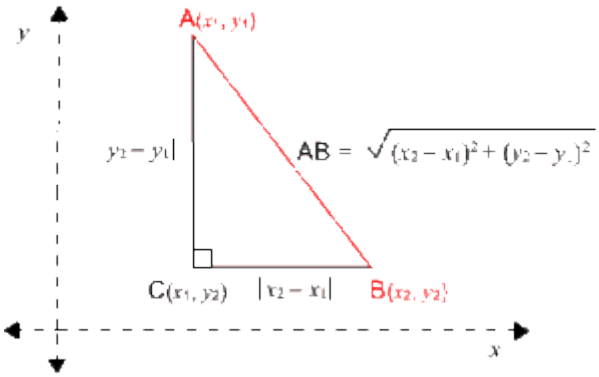

Figure 5 Distance formula

In addition, we used the formula in figure (4) for calculating the ground robot's angle and then calculating the angular speed [9].

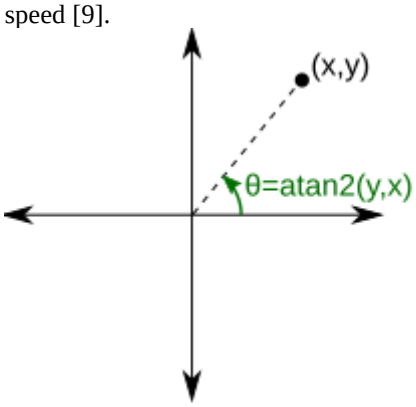

Figure 6 Angle formula

An obstacle avoidance algorithm was developed for the laser scan topic [10]. If the ground robot's laser scan detects an obstacle below the threshold by one, the robot will stop and rotate for finding a clear path. We tested the robot in an empty world to verify the goal and obstacle functions, then we integrated into the desert environment.

D. Communication

The communication between the drones and the ground robot is managed by an action server. When the target is found by one of the drones' cameras, the camera is calculating the distance to the car (target) and then transforms the measured distance into coordinates (x, y). The drones mission file is sending the target's coordinates to the action server. The ground robot will dispatch to the target once it received the target coordinates from an action server. Both drones will communicate with each other to return to the base.

E. Execution

The proposed project has three main folders, which are the drones package, the husky package, and search and rescue package. The search and rescue folder contains the action server file, UAV_mission, and UGV_mission. Figure (5) is showing the entire setup for our project, which we uploaded to GitHub for more details. [11]

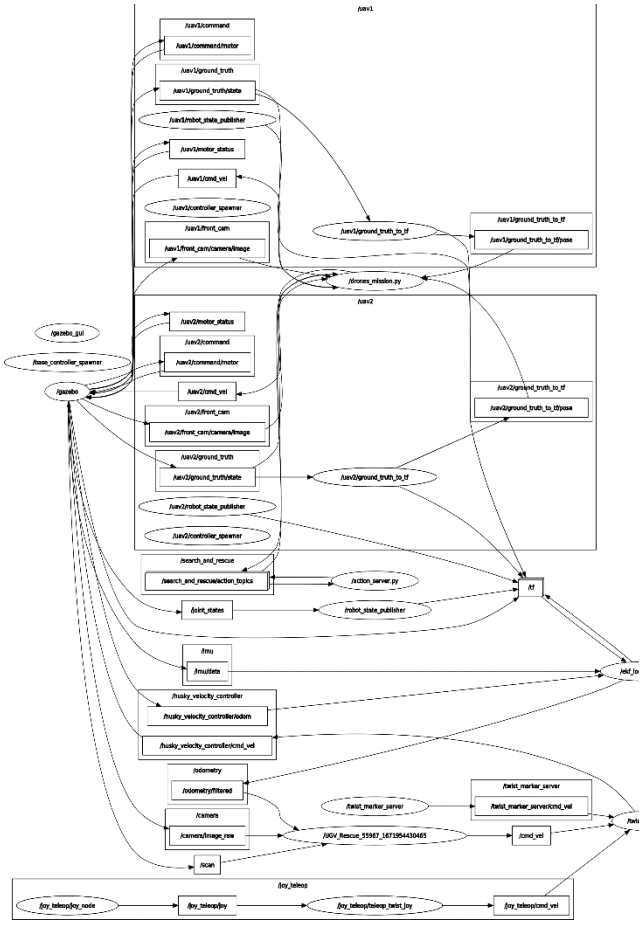

Figure 7 ROS graph

VI. Experimental Result and Chanllenges

A. UAV mission

The drones are used for scanning the area for detecting the target. The camera captured the target and process it to measure the distance as it shown in figure (8) and (9) accordingly.

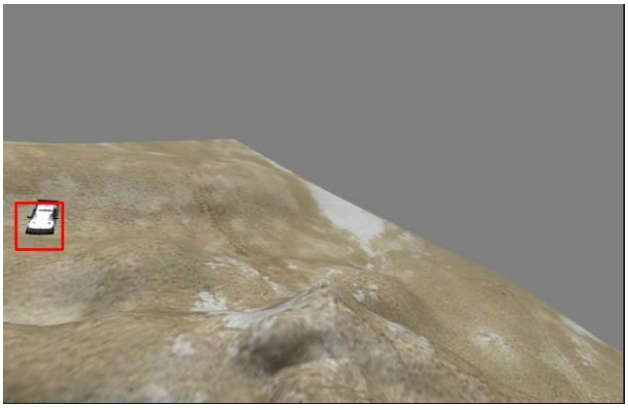

Figure 8 Target found

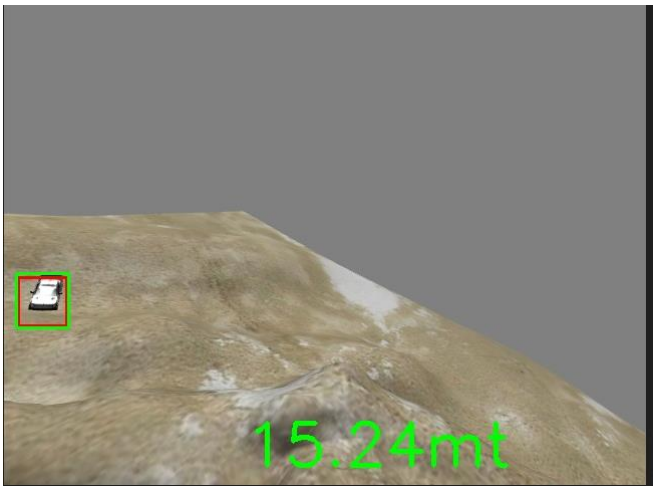

Figure 9 distance between camera & target

B. UGV mission

The husky ground robot is tested first, on empty world to verify go to goal function, figure (10) shows the robot traveled from orign coordinates (0,0) to (3,3).

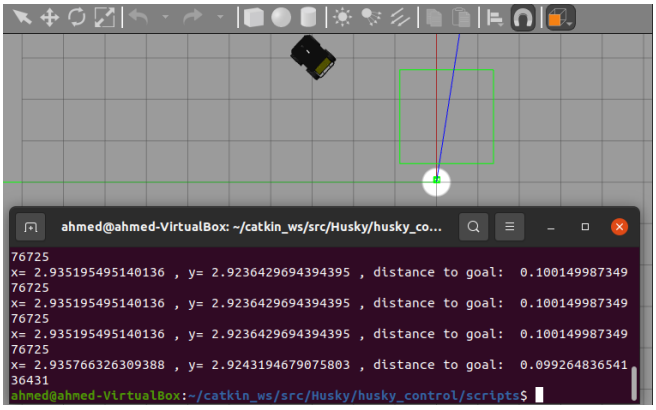

Figure 10 Husky in empty environment

In the final stage, we were able to include all the codes under one launch file.

Challenges:

During the work on the project, different issues and challenges was encountered, starting from setting up the environment, gazebo was crashing when trying different models such as cars, humans, etc.. even though some packages that contain such models has been used ,but mainly we were able the setup the required environment and move to start with the spawning the drones.

Two hector quadcopters were used, namely, UAV1 and UAV2, UAV1 was excellent and different sensors and components are working well, on the other hand, UAV2 was having issues in the camera feed, it was causing errors in the code and the imaging was discontinues and freezing, along with being behaving weirdly in the middle of the scenario execution.

While executing the scenario, multiple techniques have been used such as computer vision, obstacle avoidance, and camera to target distance algorithms, the algorithms are

generally good, however, sometimes they glitch and cause wrong results as observed during experimentations.
There are two major encountered issues:
First, Losing control on UAV2 when executing return to base function

- The control gets disconnected and the publishing commands to the drone is failing

Second, Husky-UGV gets stuck during the motion to target

- The mountain terrain environment is causing the Husky-UGV to get stuck in a specific area, even though the obstacle avoidance is helping to avoid mountains or any encountered hills, we believe it's a glitch which causes this issue.

VII. Contribution

The most crucial aspect that helps a team perform well is having a clearly defined objective that each member works toward. When each member of the team is aware of the team's direction, they may continuously offer their thoughts and insights to accomplish that objective.
The team consists of two members, Eng. Shabib Aldossary, and Eng. Ahmed Alharthi. The tasks of the proposed project are distributed between us as Shabib manages the installation of hector_quadrotor_notic, the mission of the drones by developing spiral movement code in python. In addition, the detection of the target by drones. Shabib was able to transform the distance between the camera and the target and send it to the action server as a coordinate. On the other hand, Ahmed was responsible to install the husky ground robot. Also, launching the ground robot to the goal and applying an obstacle-avoiding algorithm.
There are tasks that the team was able to accomplish together such as integrating a desert environment into the gazebo, communication between drones and ground robot, and executing the project in one launch file. Thanks to Shabib for uploading the project setup to GitHub. [11]

VIII. Conclusion

The proposed project was implemented for a noble reason, saving lost people in the desert. The team at the early stage of the project was identifying the use case and the activity diagram. The team used ROS noetic and a gazebo for reflecting their idea in a simulation environment. The proposed simulation project is using two hector quadrotors (drones) and one husky (ground robot). When both drones launch and start scanning for the target, which the team used a car in the desert area. If one of the drones detects the target, images will be processed to measure the distance between drone's onboard camera and the target, and send the target's coordinates to the action server. The action server will send the receiving coordinates to the husky ground robot. The husky will dispatch to the target. The team was able to implement their idea and they have different issues and challenges.

IX. References

[1] S. Salama, "https://gulfnews.com/world/gulf/saudi/saudi-arabia-131-people-went-missing-in-desert-last-year-1.78403752," 8 April 2021. [Online].

[2] [Online]. Available: https://files.fm/u/8qwbk26ss.

[3] [Online]. Available: https://github.com/Tinker-Twins/Husky.

[4] [Online]. Available: https://www.ros.org/.

[5] [Online]. Available: https://gazebosim.org/home.

[6] [Online]. Available: https://www.academia.edu/60848563/Tsunami_post_disaster_robot_Simulation_and_implementation_of_aerial_path_planning_to_explore_unknown_environment.

[7] [Online]. Available: http://wiki.ros.org/ROS/Tutorials.

[8] [Online]. Available: https://www.toppr.com/guides/maths/coordinate-geometry/distance-formula/.

[9] [Online]. Available: https://en.wikipedia.org/wiki/Atan2.

[10] [Online]. Available: https://github.com/Rad-hi/Obstacle-Avoidance-ROS.

[11] [Online]. Available: https://github.com/UtterlySus/Search_and_Rescue.