S.A.R.A

Search And Rescue Assistant

Software Engineering | Group 8

Technical Documentation

https://abhiek187.github.io/emergency-response-drone/

TEAM MEMBERS

Sahana Asokan
Won Seok Chang
Avnish Patel
Abhishek Chaudhuri
Shantanu Ghosh
Srikrishnaraja Mahadas
Sri Sai Krishna Tottempudi
Vishal Venkateswaran

Table of Contents

1. Introduction

Our goal of the project is to reduce the loss of life caused by fires every year, by having a drone with multiple features (thermal sensor, overhead viewing, etc) that the first responders will use through our user interface. With S.A.R.A's easy-to-use interface, we can make sure that first responders will have a working system without any difficulties. S.A.R.A will be equipped with 2 ultrasonic sensors on the system that will assist the user when it comes to obstacle detection. Some of its other features will help locate people who need help. One such example is the thermal camera. It can detect heat signatures through materials to let the responders know where the victims are located or if there is any heat on the premises. Utilizing the drone will reduce the number of lives that are endangered as first responders will no longer need to explore every room in order to find people trapped inside the building.

2. Website

The controller for the drone is rendered in HTML on our website and uses CSS to layout the buttons and drone data in a user-friendly way. The video feed of the camera is located in the center. The website grabs info about the user's device and starts streaming video through the *navigator.mediaDevices* library. The video source can be selected via a dropdown menu at the top left. If no video can be shown or if the user doesn't allow the webpage to use their camera, the user will be presented with either a black screen or text saying "Drone Camera Offline". The user can toggle the visibility of the camera by pressing the power button on the top right.

At the bottom are data values related to the drone. To the left is data showing the device's battery level (only supported in Chrome and Opera), the speed of the camera, and the device's current location. Note that the user must allow the device to get their current location for these features to work. The battery level is retrieved through *navigator.getBattery()* (now obsolete). It will warn the user about low battery if the battery level drops below 20%. The location is gotten through *navigator.geolocation* with coordinates rounded to the 5th decimal place. The speed is then calculated using the coordinates every time the location updates.

Over to the right is the sensor data, which is picked up by the 2 ultrasonic sensors. Both these measurements are being fetched from a text file located within the website's filesystem. Every time the text file updates, it will be reflected on the website. The text will turn red if an object is within a foot of the sensors (~ 30 cm) to indicate that the user should steer away from the obstacle. More information about the sensors can be found in the subsequent sections of this document.

3. Obstacle Detection

Obstacle detection is performed primarily by the ultrasonic sensor named HC-SR04. This sensor will be placed in 2 different areas of the drone: the left and right side of the drone. The sensors will be placed in these locations for optimal drone safety and obstacle detection. The ultrasonic sensor is utilized through a Raspberry Pi. The Pi allows the user to measure the distance calculated by the Pi through a Python script, and send it to the user via a text file. The HC-SR04 sensor works through its two main components on the sensor. The trigger, which is responsible for sending a pulse or chirp towards the object that the drone needs to detect, and the Echo, which receives the reflected pulse from the object. The calculation that the Pi makes involves the speed of sound and the time it takes for the Echo to read the pulse sent by the trigger. Since we know the basic physics equation v(velocity) = d(distance)/t, and that the pulse travels 2*d because of the distance to and from the measured object, we can find the distance between the object and the drone. Therefore the distance is equal to (v*t)/2. This distance is measured in cm. The range of HC-SR04 is anywhere from 3 cm to 3.5 m.

4. Operating the drone

- The way the drone will work is that firefighters will deploy the drone and will be able to maneuver it using the remote controls.
- There will be a monitor for them to see what the drone is seeing. The monitor will also pin out the location of survivors on the floor plans. This will guide the firefighters in the process of detecting survivors and the dangers in the building.
- Turn on the power
- Connect the sensors onto a breadboard as shown in the PIN_TRIGGER and PIN_ECHO variables in the Python file and attach the breadboard to the base of the drone.
- Make sure the camera and all the sensors are working
- Will show the current location and will change as you move around
- Make sure the speedometer is working
- Test the controls of being able to maneuver
- Finally, if everything is working, the drone is ready for use and help save lives

5. How to Run

1. Setup the drone. (as seen in operating the drone)
2. Connect a mobile device (preferably an Android phone) to the drone's base.
3. Using a laptop, connect to the phone locally (such as with SideSync) and open the drone controller on any browser. Google Chrome is recommended to ensure most functions work as they should.
4. Adjust the camera so it views the drone from its front side.
5. Power the drone on.
6. Start flying and help people in need!