Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

Team 36: Self-Navigating, Obstacle Avoiding Robot
Bi-Weekly Update 4

Teammates:

Arkadi Zhanov

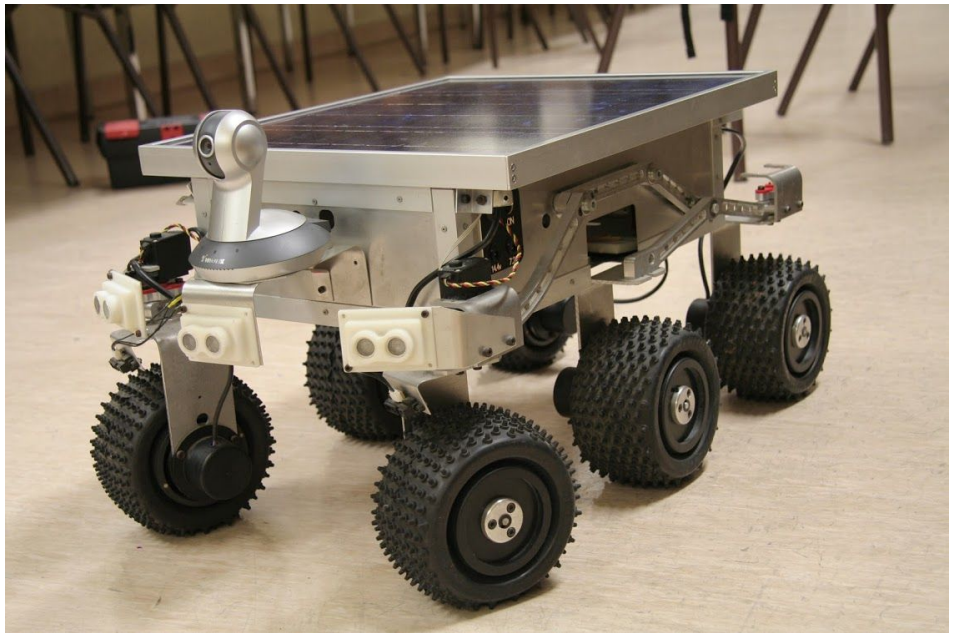
Nathan Sommer

Nikolai Paderin

Sponsor: Stavros Kalafatis

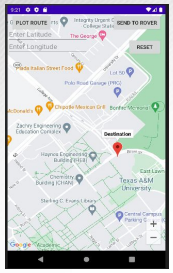
Project Summary

- Create a system that takes in a point on a map as the rover's destination, creates a route for the rover, and gets the rover to its destination and back while avoiding obstacles.
- The main motivation for this project is for application in military settings such as aid delivery to wounded soldiers in the battlefield or local, residential delivery of food or mail.



Integrated System Diagram

App



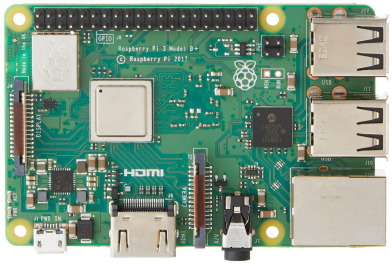
Receives Location and Route

Sends User Destination

4G HAT

Sends Data

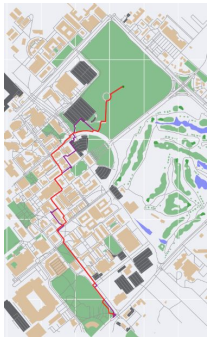
Raspberry Pi



Receives Route

Sends Location and Destination

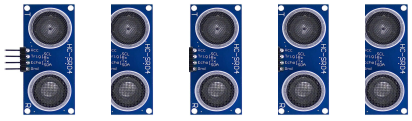
Route Creation (On Raspi)

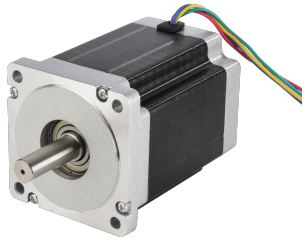


-87.615	41.83317
-87.615	41.83317
-87.615	41.83317
-87.615	41.83317
-87.615	41.83317
-87.615	41.83317
-87.615	41.83317
-87.615	41.83317
-87.615	41.83317

Override navigation movement control and use obstacle avoidance movement

Obstacle avoidance sensor array detects nearby obstacles





Control motor movement based on current position and next node position

Project Timeline

(Green done, light green near complete, yellow underway, red in trouble, white not started)

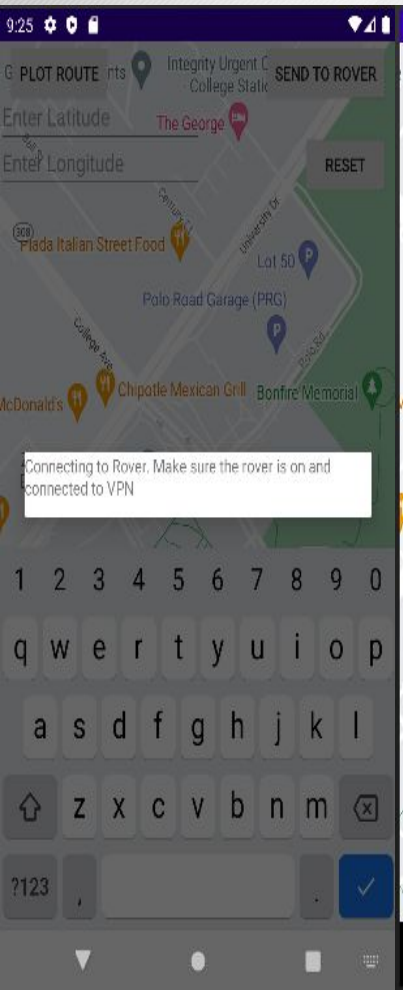
Subsystem Designs and Testing (completed 9/11)	Integrate Pathfinder and Movement/Controls and test (to complete by 9/20	Integration with Android App and Raspberry Pi using cellular data (to complete by 10/5)	Final Integration (to complete by 10/15)	Systems Test (to complete by 11/2)	Validation (to complete by 11/26)	Demo and Report (to complete by 12/5)

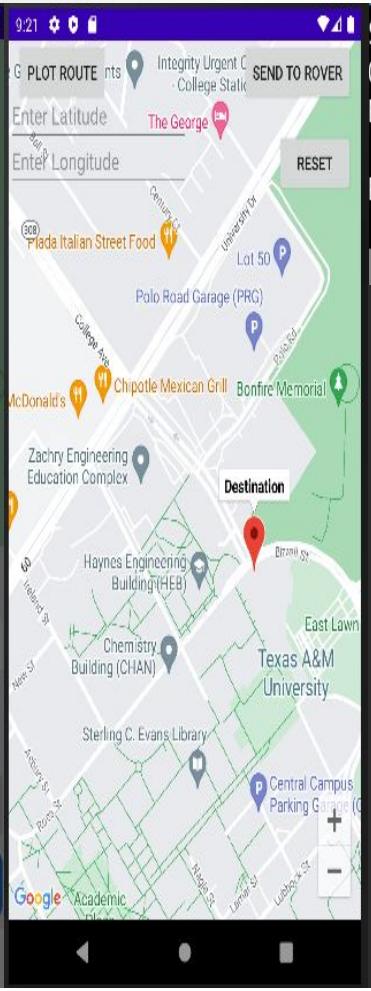
*Need to Install Libraries

Nathan Sommer

Accomplishments since last update 13 hrs of effort	Ongoing progress/problems and plans until the next presentation
Integration(App-Pi) Able to send and receive information over different networks through OpenVPN, as well as update the app UI when given GPS	- Help with other subsystem's connections - Implement Kill Switch - Socket Error Handling (Low Priority) - UI Updates(Low Priority)

Nathan Sommer






```
Server listening on 100.96.1.4:12345
Connection from ('100.96.1.2', 43369)
Received from client: 30.620109859811027 -96.3381339982152

Received from client: 30.620426661206437 -96.33994851261379
```

```
public String[] keys = {"q123w21", "qei3uUs3-r", "a1s2f3gte4"};
1 usage
public int[] portnum = {12345, 11111, 11112};

1 usage
public String[] ipaddress = {"100.96.1.38", "100.96.1.39", "100.96.1.40"};
```

```
if (in.ready()) {
    str = in.readLine();
    System.out.println("Server Says: " + str);
    String[] numberStrings = str.split( regex: "\\s+");
    double GPSLat = Double.parseDouble(numberStrings[0]);
    double GPSLong = Double.parseDouble(numberStrings[1]);
    runOnUiThread(new Runnable() {
                      @Override
                      public void run() {
                     //Add a marker on the map
                      if (mMap != null) {
                          LatLng markerLocation = new LatLng(GPSLat, GPSLong);
                          mMap.addMarker(new MarkerOptions().position(markerLocation).title("Marker Title"));

                     // Move the camera to the marker's location
                         mMap.moveCamera(CameraUpdateFactory.newLatLng(markerLocation));
                             }
                         }
                      });
}
```

Arkadi Zhanov

Accomplishments since last update 16 hrs of effort	Ongoing progress/problems and plans until the next presentation
Integration (Pathfinder-Pi) - Identified integration issues (library version incompatibilities) - Pathfinder subsystem is complete and is nearly entirely integrated with other subsystems **Integration (Obstacle Avoidance-Pi)** - Ultrasonic distance sensor readings tested and validated - Developed two obstacle avoidance approaches (vector weights / cases) for controls integration - Obstacle avoidance code tested, and mostly validated and integrated	- Install specific library versions on microcontroller - Finalize deployment, testing and validation of pathfinder system - 3D print a more stable housing for ultrasonic sensors - Pick the better obstacle avoidance procedure - Finalize testing and validation of obstacle avoidance system - Help with other subsystems

Arkadi Zhanov

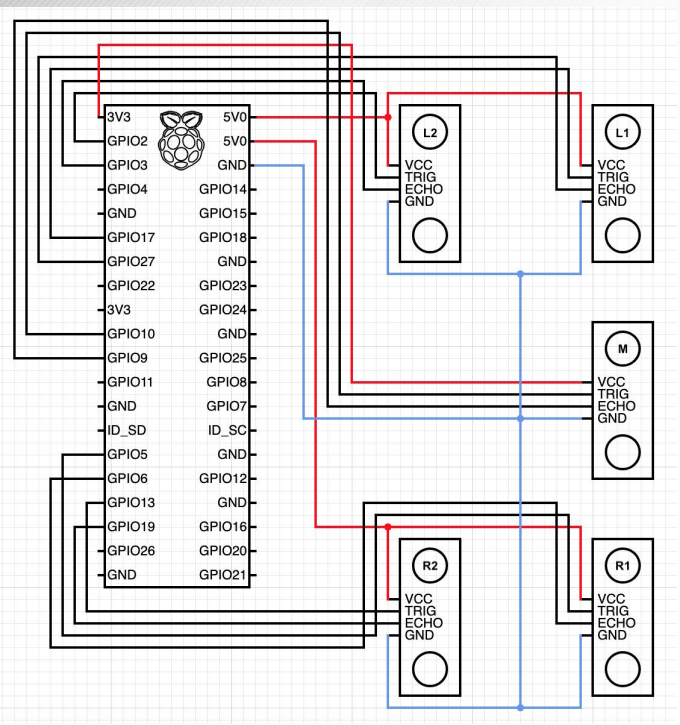


Fig 1. Current Wiring Diagram of Obstacle Avoidance Sensor Array and Microcontroller

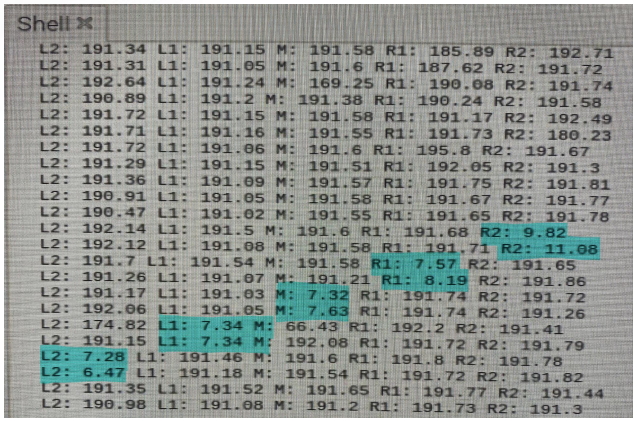


Fig 2. Testing and Validating Distance Sensor Readings Before Combining With Obstacle Avoidance

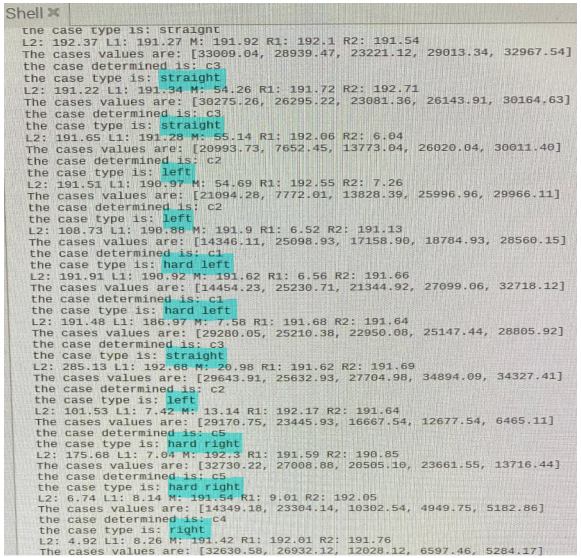


Fig 3. Obstacle Avoidance Testing and Validation

Navigation/Movement/Control

Nikolai Paderin

Accomplishments since last update 25 hrs of effort	Ongoing progress/problems and plans until the next presentation
4G Hat -GPS functionality -Able to connect to the 4G cellular network **Integration(Pi - App)** -Able to send and receive information to and from the application. (tested) -Able to use cellular network to access the internet, enabled on boot up. **Integration(Pi-Navigation)** -Loaded the Navigation code to the pi.	**GPS** -Is currently able to pull information, but has issues with booting up. **Motor Control** -Test functionality with GPS module being used simultaneously. (They both use the same serial port, may have to switch to Pi 4)

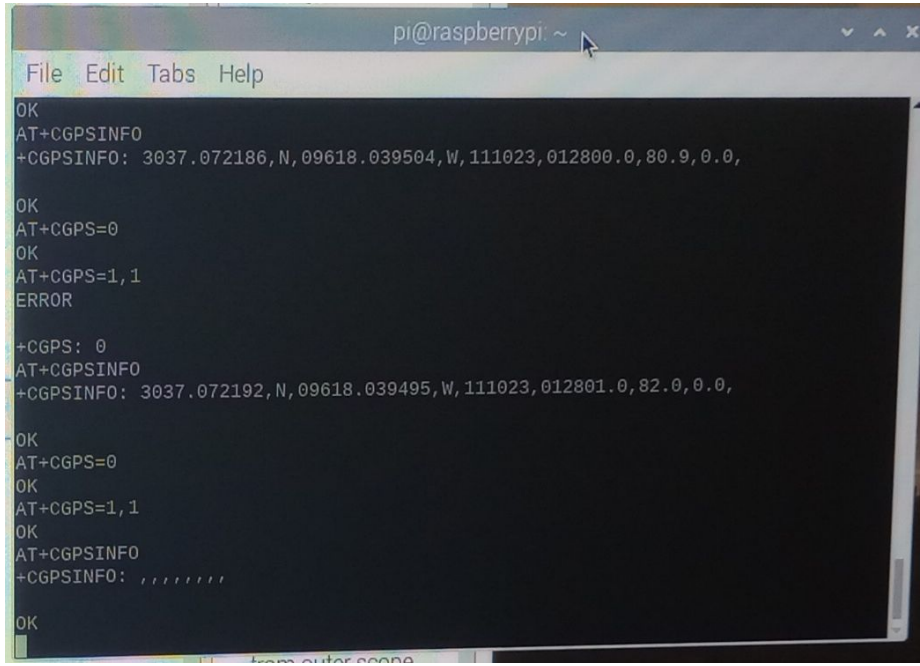
Navigation/Movement/Control

Nikolai Paderin

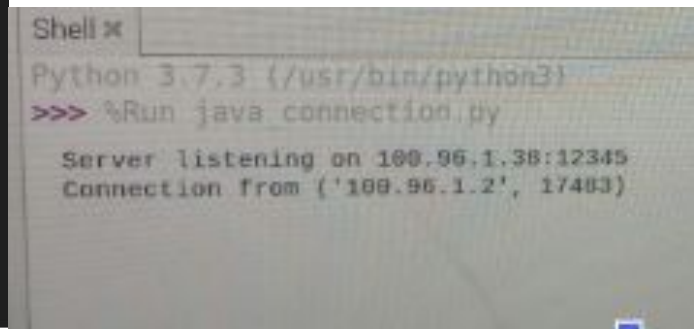
Able to connect to server and send info

+

Able to request GPS location




```python
import socket

# Create a socket object
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to a specific address and port
host = "hostip"
port = 12345
server_socket.bind((host, port))

# Listen for incoming connections
server_socket.listen(5)

print(f"Server listening on {host}:{port}")

while True:
    # Accept a connection from the client
    client_socket, addr = server_socket.accept()
    print(f"Connection from {addr}")

    # Send a message to the client
    # message_to_send = "Hello, client!"
    # client_socket.send(message_to_send.encode('utf-8'))

    # Receive data from the client
    while True:
        data = client_socket.recv(1024).decode('utf-8')
        print(f"Received from client: {data}")
```



Execution Plan

SYSTEM Validation Plan

TEST NAME	REQUIREMENT	SUCCESS CRITERIA	METHODOLOGY	STATUS	ENGINEER RESPONSIBLE
403 Cleanup Nikolai	FSR Pg10	Have all sensors + movement code working and precise	Test sensors, and rover movement by measuring and calibrating	Complete	Nikolai
403 Cleanup Nathan	FSR Pg10	Add the Connection Page + Help Arkadi with navigation subsystem	Test to see if subsystem is fully operational and working as intended	Complete	Nathan
403 Cleanup Arkadi	FSR Pg10	Have the navigation/pathfinder code working as intended.	Using two points create a series of nodes that connect them using the code	Complete	Arkadi
Integrate Pathfinding with movement	FSR Pg10	Have the Raspberry pi communicate with pathfinding code and integrated	Using the Pi, send info to the pathfinding subsystem to create a path	Complete	Nikolai + Arkadi
4G hat connection + integration	FSR Pg10	Have the Raspberry pi integrated with a 4G hat such that it has internet access wherever it is.	Using the Pi and the 4G hat, give the Pi internet access through a cellular network	Complete	Nikolai
App to Raspi connection	FSR Pg10	Have the Raspberry pi send and receive information from the App	Test sending information with Raspi and App using sockets	Complete	Nathan + Nikolai
Demo Test/Validation	FSR Pg10	All subsystems are successfully talking to each other, and can operate autonomously	Use app to get user destination, and see if rover goes to destination using all subsystems	Not Complete	Nathan + Nikolai + Arkadi
Systems Test	FSR Pg10	System is working as intended	Stress-Check System, and see if everything works as intended	Not Complete	Nathan + Nikolai + Arkadi

THANK YOU FOR YOUR TIME!

ANY QUESTIONS?