Team 16 MDR
Trash·*E*

University of Massachusetts Amherst
BE REVOLUTIONARY™

Meet The Team

Stephen Townsend
Computer Engineer

Jasmine Hickey
Electrical Engineer

Smit Patel
Computer Engineer

John Diep
Computer Engineer

Team 16 Advisor
Professor Do-Hoon Kwon

PROBLEM STATEMENT

Bringing garbage to the curb can be a difficult task for some, whether they are physically disabled or elderly. Aid lent by caring companions can be unreliable as everyone has their own schedule and may not be available at the necessary time. Those in need of refuse removal can be assisted using *Trash·E*. *Trash·E* will take the strenuous part of taking the barrel to the curb out of the process; it is a Wi-Fi enabled robotic system to move garbage from its near-home collection site to the curb on a path and at a specified time.

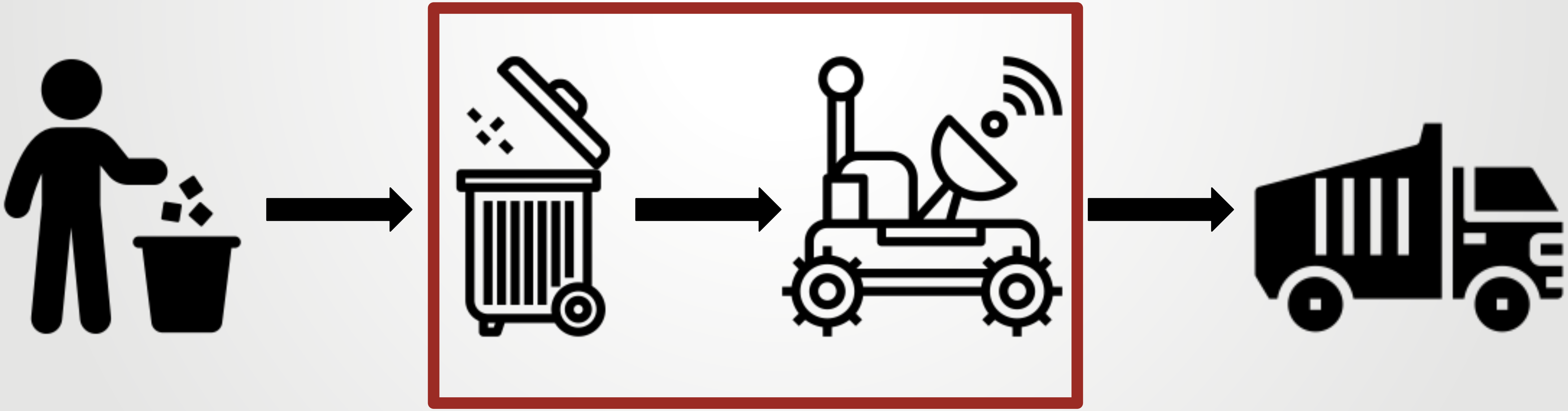
BE REVOLUTIONARY™

DESIGN GOALS, SPECIFICATIONS, AND TESTING PLAN

University of Massachusetts Amherst

GOAL

The goal of the *Trash·E* is to provide an automated and convenient way to move trash from its place of collection to the location needed for removal by a trash service with minimal user intervention.



What has changed since PDR?

- **Navigation System**
  - ~~Stepper motors~~
  - Line tracking
  - GPS
  - IMU
- **User Interface**
  - ~~Route preset~~
  - Scheduling
  - Obstacle alerts
- **Prototyping Phase**
  - Small tabletop model
  - Scale up for CDR
- **Power Supply**
  - Ground based power for MDR

- **Why did we make these changes?**
  - Guidance from course coordinators
  - Chris Caron (TA)
  - SDP Scope

Design Goals

- The System shall
  - specify what days and times *Trash·E* will run
  - wait for a preset amount of time at the pickup location before returning to its starting location
  - detect when trash is picked up
- Navigation
  - follow designated line forward and backwards
  - return to line if deviation occurs
- Obstacle Detection
  - detect and stop before obstacles in front of the system
  - alert the user of an obstacle
- Environmental Restrictions
  - fully functional on paved surfaces

System Specs (Updated)

Requirement	Specification	Value
Arrive before specified pickup time	User-inputted day/time	Arrive 10 minutes before time specified, alert user
Payload of average trash bag	Pounds	Greater than or equal to 25 lbs.
Battery Life	Trips / Time	Two round trips on one full charge
Wait at Pickup Location	Time / Load Verification	Wait until load is take away or EOD
Object Detection	Distance	Detect objects <= 1 meter from rover (90% accuracy)
Navigation [Primary]	Line Tracking	Follow line from start to pickup and back (90% accuracy)
Navigation [Secondary]	Line Finding	Find line after disturbance or error within five minutes (80% accuracy)
Movement	Surface, Slope	Work on paved surfaces up to 10° slope
WiFi Connectivity	Distance	20 meters, LOS
Arrive at Starting & Pickup Location	Error Tolerance	Arrive within 2 meters of final destination (90% accuracy)
User Alert [Obstacle]	Time	Alert the user of an obstacle within one minute (90% accuracy)

Testing Plan

Navigation [Primary]

- Design test track with varying designs (straight, curvy, angled, etc.)
- Designate start and pickup locations
- Test 10 times per track
- Measure distance from pickup location, emulate trash pickup
- Measure distance from start location
- Repeat for different layouts
- Repeat process outdoors

Object Detection

- Design test track of varying designs
- Place obstacle on line, mark location
- Run Trash-E on the track, mark where system stops
- Begin timer when object detected
- End timer when alert is received
- Measure distance between obstacle and rover
- Repeat 10 times

Load Detection

- Obtain varying weights up to 25 lbs
- Place weight on sensor
- Remove
- Verify detection
- Repeat 10 times

Navigation [Secondary]

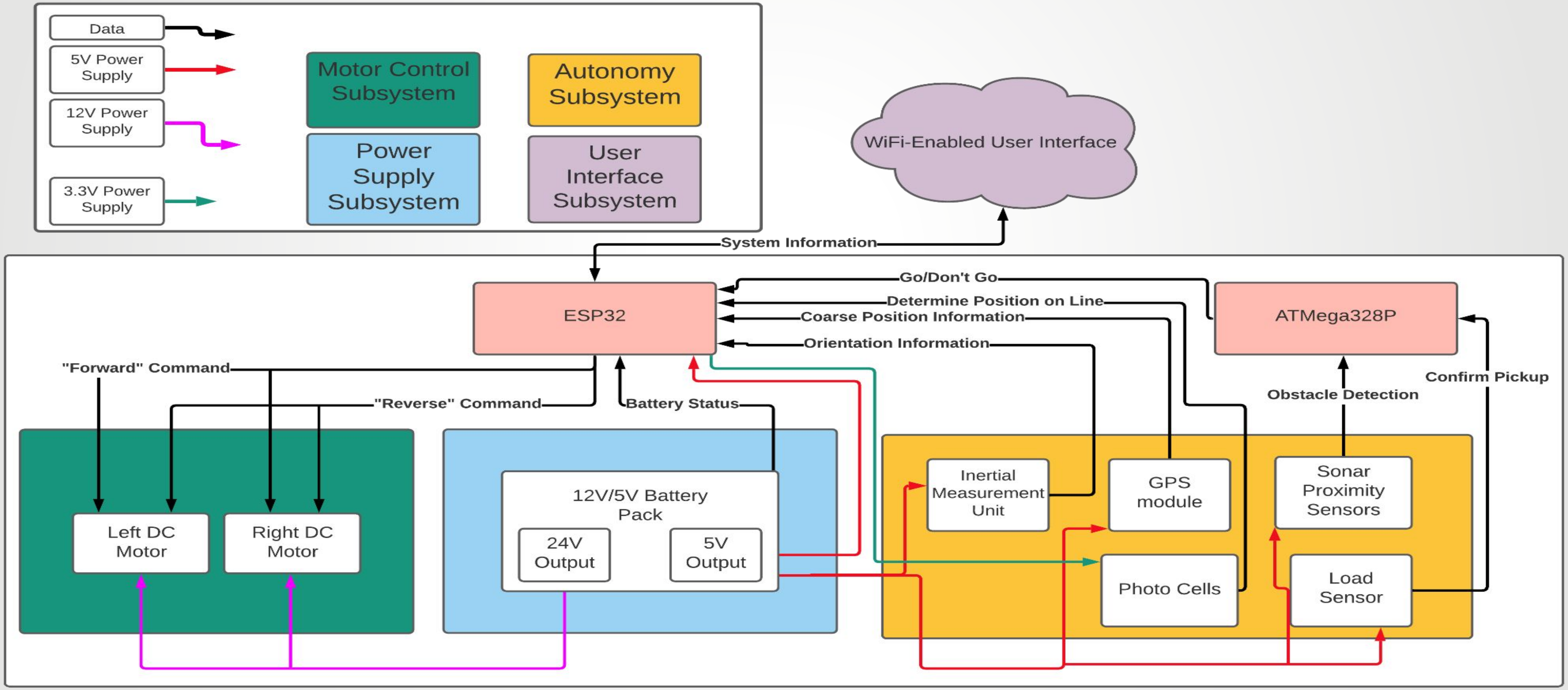
- Trash-E follows line correctly, stores data
- Emulate outside force (place rover off line)
- Start timer
- Wait until rover returns to line
- Stop timer, verify under 5 minutes
- Repeat 10 times

BE REVOLUTIONARY™

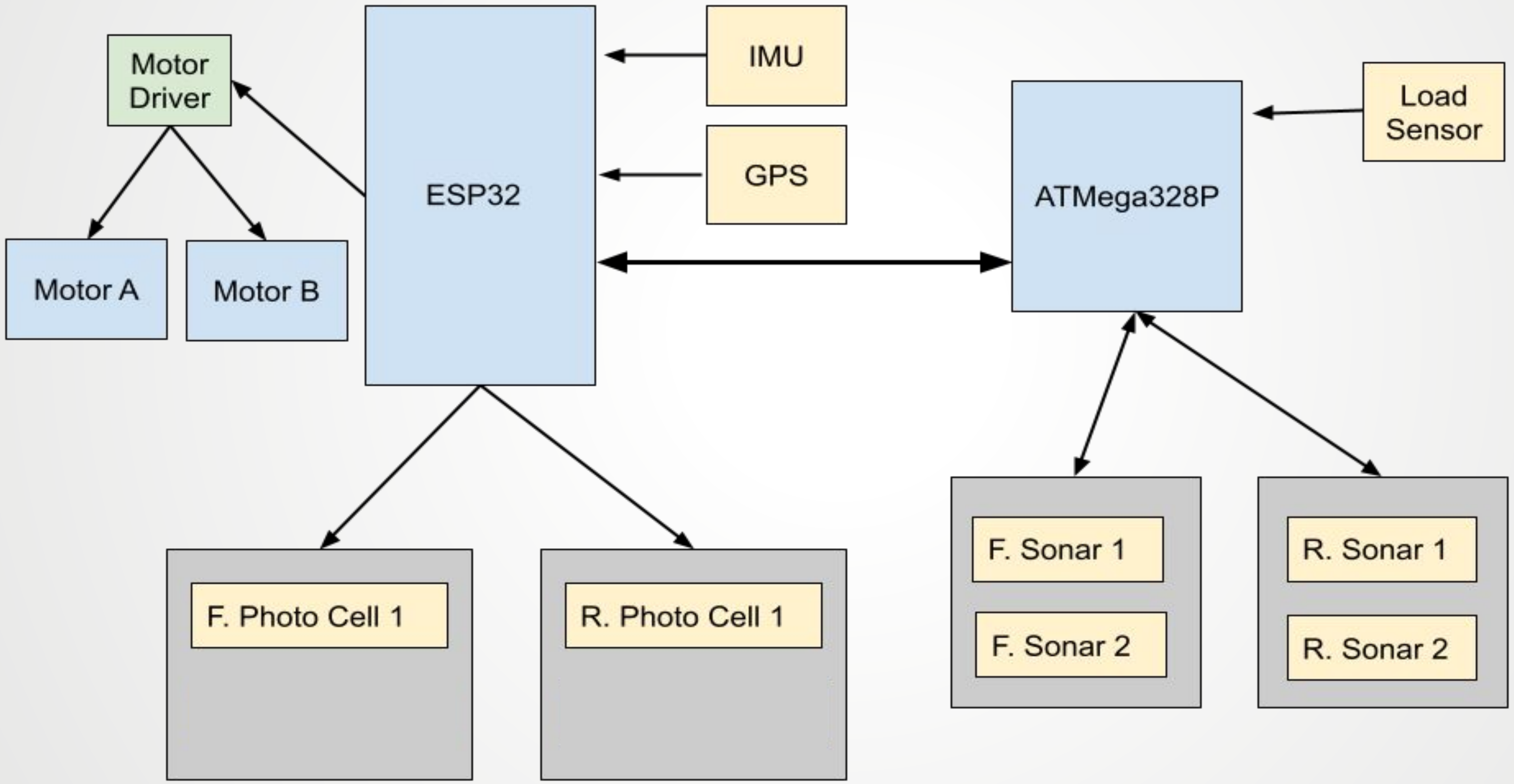
BLOCK DIAGRAMS & PCB DESIGN

University of Massachusetts Amherst

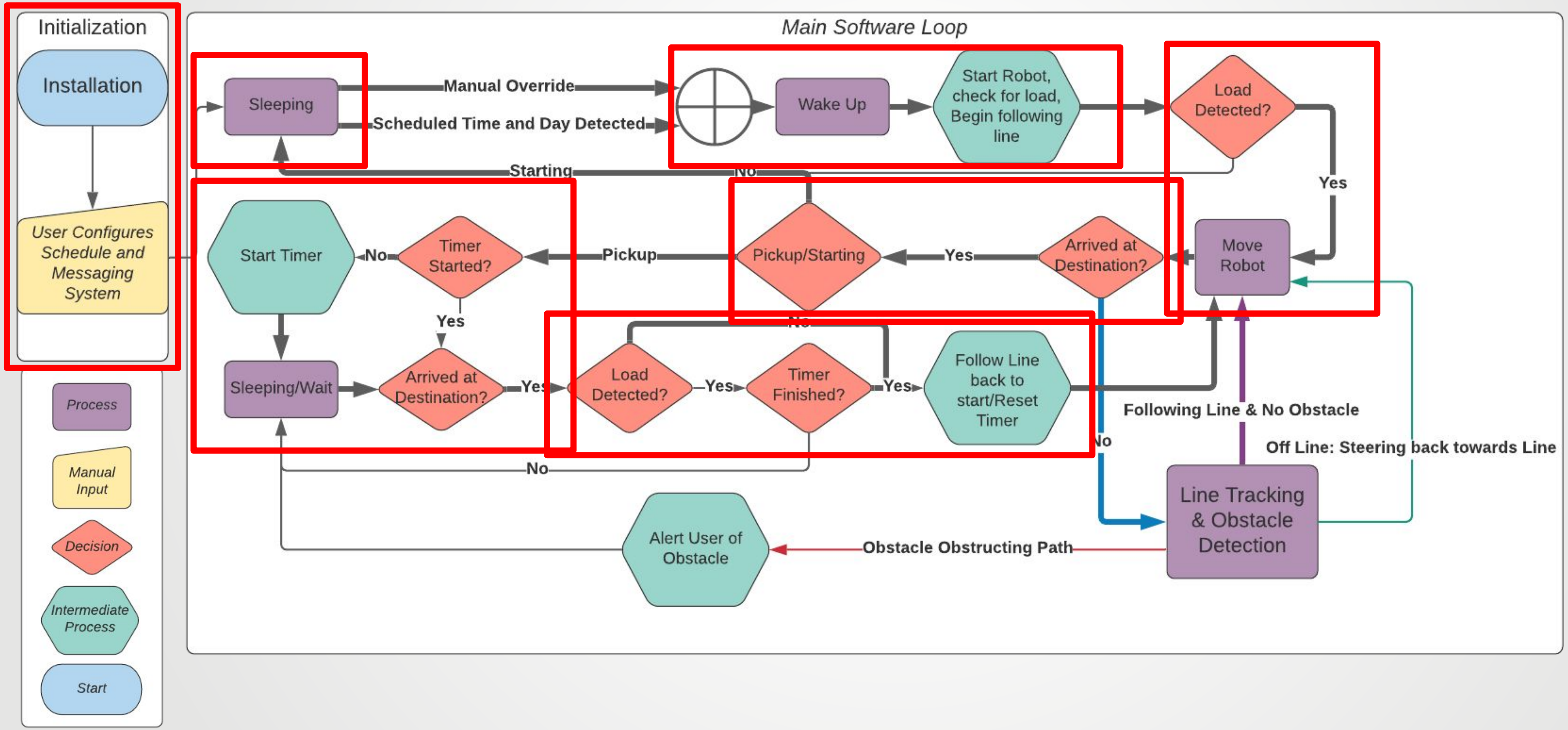
(UPDATED) SYSTEM BLOCK DIAGRAM



Hardware Block Diagram



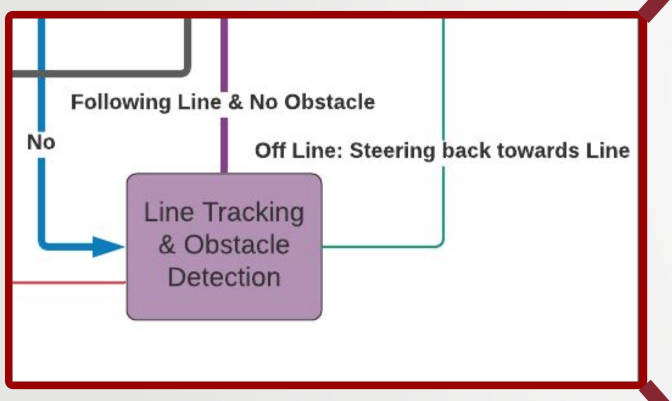
Software Block Diagram

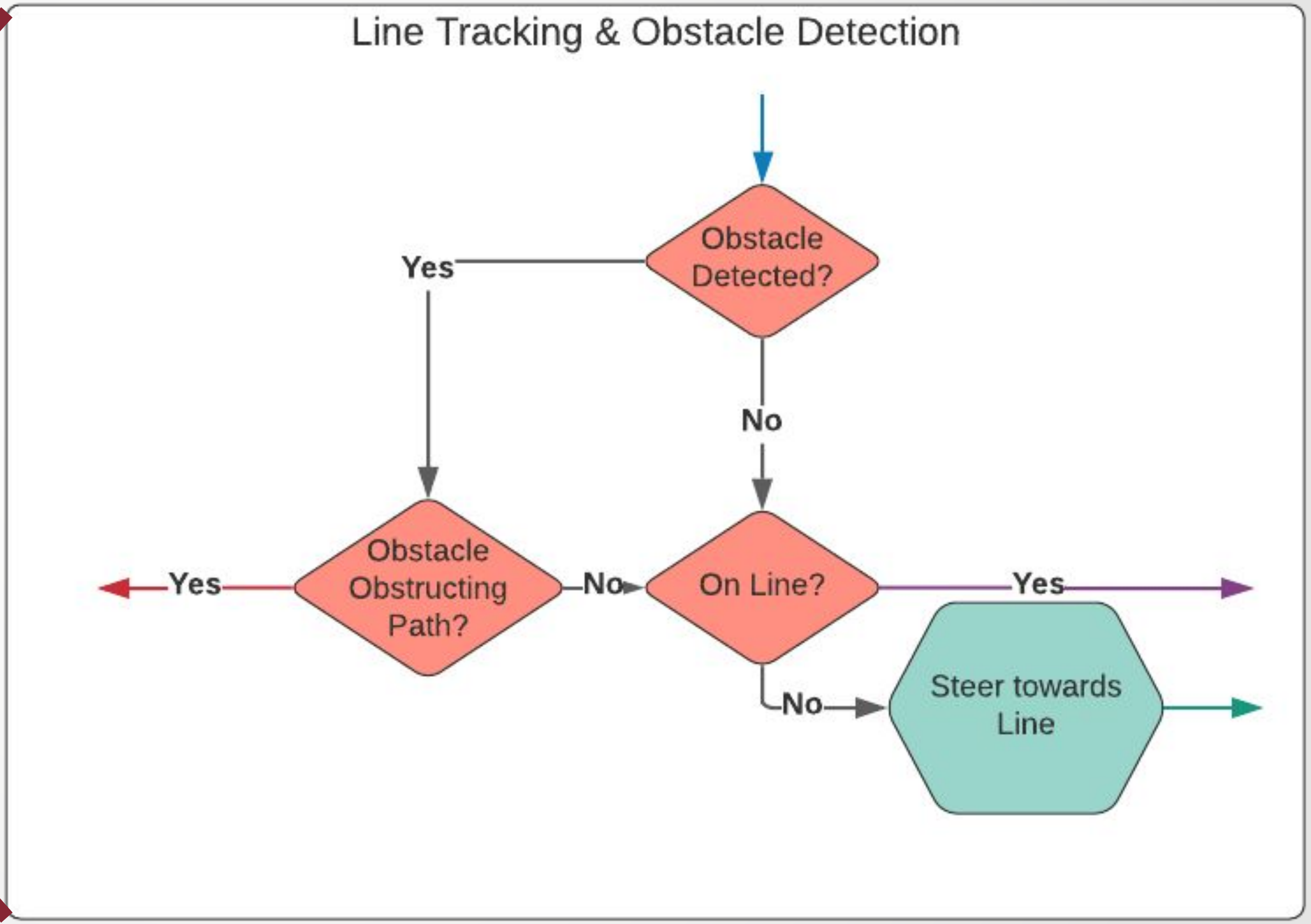


In **Bold** represents the main software path, if there were no obstacles

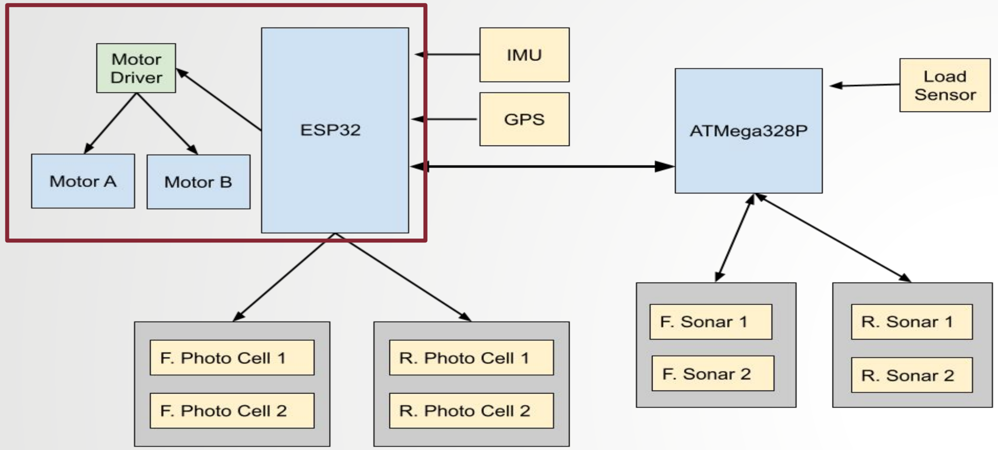
University of Massachusetts Amherst BE REVOLUTIONARY

Deeper Look: Navigation and Obstacle Detection





Subsystem Analysis: Motor Control




```
void motor_move(uint8_t highPin, uint8_t lowPin, uint8_t PWM, int spd)
{
  analogWrite(PWM, spd);
  digitalWrite(highPin, HIGH);
  digitalWrite(lowPin, LOW);

}
```

```
void forward(int speed){
  motor_move(ain1, ain2, pwmA, speed);
  motor_move(bin2, bin1, pwmB, speed);
}
```

```
void forwardLeft(int speed, int size){
  motor_move(ain1, ain2, pwmA, (speed/size));
  motor_move(bin2, bin1, pwmB, speed);
}
```

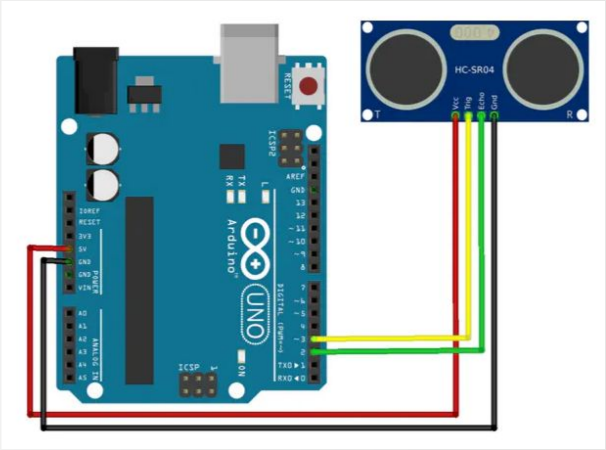
Motor Driver Logic Overview

In1	In2	PWM	Out1	Out2	Mode
H	H	1-255	L	L	Short Brake
L	H	1-255	L	H	CCW
L	H	0	L	L	Short Brake
H	L	1-255	H	L	CW
H	L	0	L	L	Short Brake
L	L	1-255	OFF	OFF	Stop

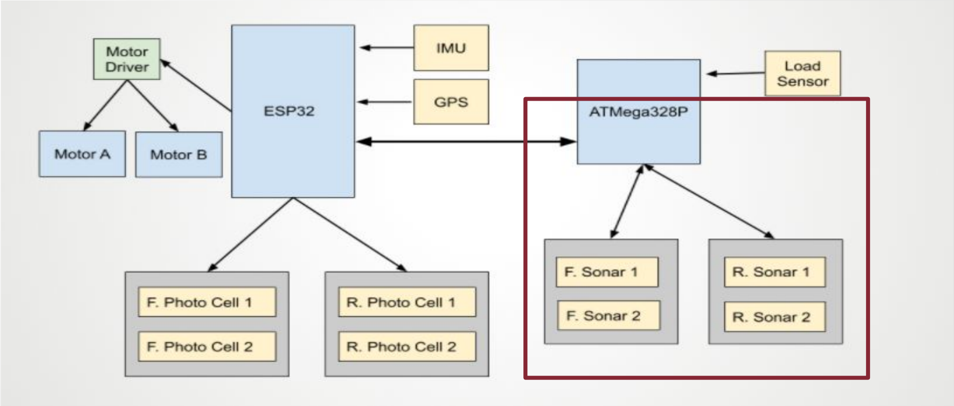
Subsystem Analysis: Sonar Sensors

```
bool direction = true; // true = forward / false = backward
```

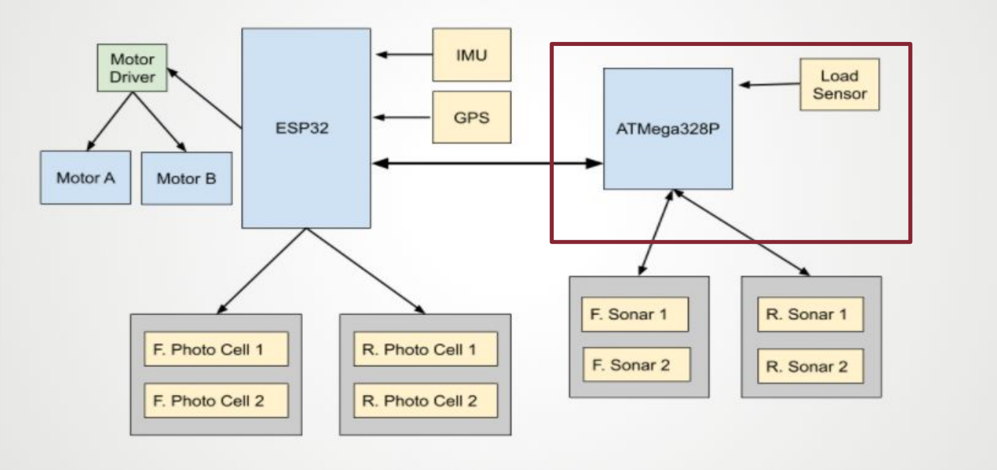
```
if (direction) { // if we are facing forward
  if (SonarSensor(trigPin, echoPin, distanceThreshold, led) || SonarSensor(trigPin2, echoPin2, distanceThreshold, led2)) {
    digitalWrite(obstacleLED, HIGH);
  }
  else {
    digitalWrite(obstacleLED, LOW);
  }
}
else {
  if (SonarSensor(trigPin3, echoPin3, distanceThreshold, led3) || SonarSensor(trigPin4, echoPin4, distanceThreshold, led4)) {
    digitalWrite(obstacleLED, HIGH);
  }
  else {
    digitalWrite(obstacleLED, LOW);
```

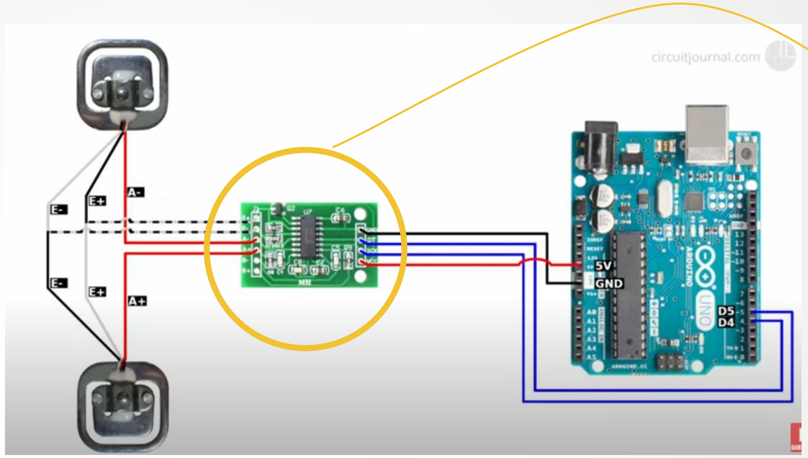


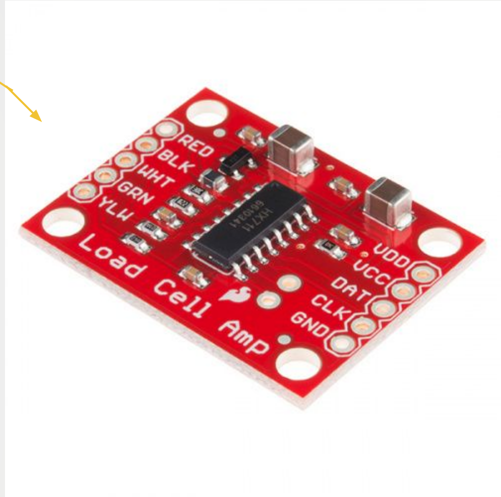

```
if(distance <= threshold) //if distance is less than 10 cm
{
  if (LED >= 0) {
    digitalWrite(LED, HIGH);
  }
  return true;
```



Subsystem Analysis: Load Sensors








```
scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
scale.set_scale(calibration_factor); //This value is obtained by using the SparkFun_HX711_Calibration sketch
scale.tare(); //Assuming there is no weight on the scale at start up, reset the scale to 0
```

```
if (scale.get_units() > 150) {   //150 grams ~ .3 lbs
 digitalWrite(10, HIGH);
  }
else {
     digitalWrite(10, LOW);
    }
```

HX711 load cell amplifier:

- gets measurable data from load cells

Subsystem: Line tracking via Photoresistor

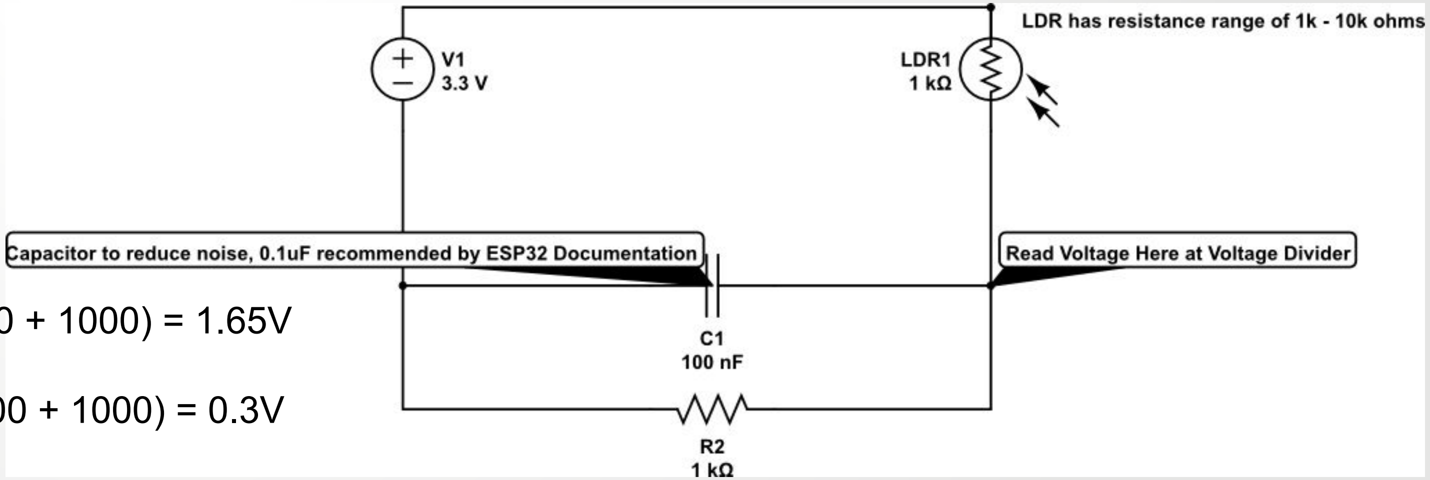
Step by Step Process:

1. Calibrate Photoresistors
   a. Find brightest color -> highest voltage value
   b. Find darkest color -> lowest voltage value
2. Perform Analog Read via ADC on voltage divider
3. Squeeze newly read value into our calibrated range
   a. Convert to a boolean value
4. Translate Boolean Values into human understandable cases
5. Repeat steps 2 - 4

```
int readRawPhotoVal(uint8_t photoCellPin) {
  /*
   * Singular version of readRawPhotoCells, does the same thing except only for one photocell.
   */
  return analogRead(photoCellPin);
}

void readRawPhotoCells(uint8_t LeftPhotoresistor, uint8_t RightPhotoresistor, photoVals *photoValues) {
  /*
   * Fair self-explanatory: Reads raw data from both of our photo cells and stores into our photoVals struct which we have a pointer to.
   */
  if (calibrated) {
    if (photoValues == NULL) {return;}
    (*photoValues).Left_Photo = readRawPhotoVal(LeftPhotoresistor);
    (*photoValues).Right_Photo = readRawPhotoVal(RightPhotoresistor);
  }
}
```

```
if ((Left_Pho
  if (Left_Ph
    return 1;
  }
  else { //Ri
    return 2;
  }
}
else if (!Lef
  return 0;
}
```

```
: detected on one side
```



$$V_{out} = V_{in} \cdot \frac{R_2}{R_1 + R_2}$$

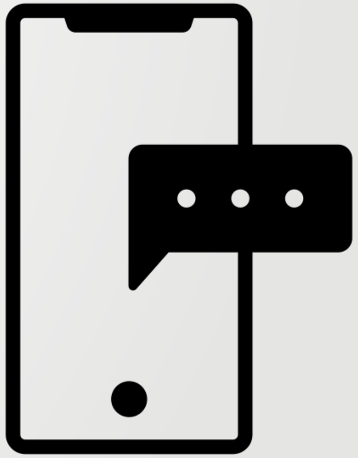
$V_{upper\ out}$ = 3.3 * 1000 / (1000 + 1000) = 1.65V

$V_{lower\ out}$ = 3.3 * 1000 / (10000 + 1000) = 0.3V

Subsystem: Messaging System on ESP32

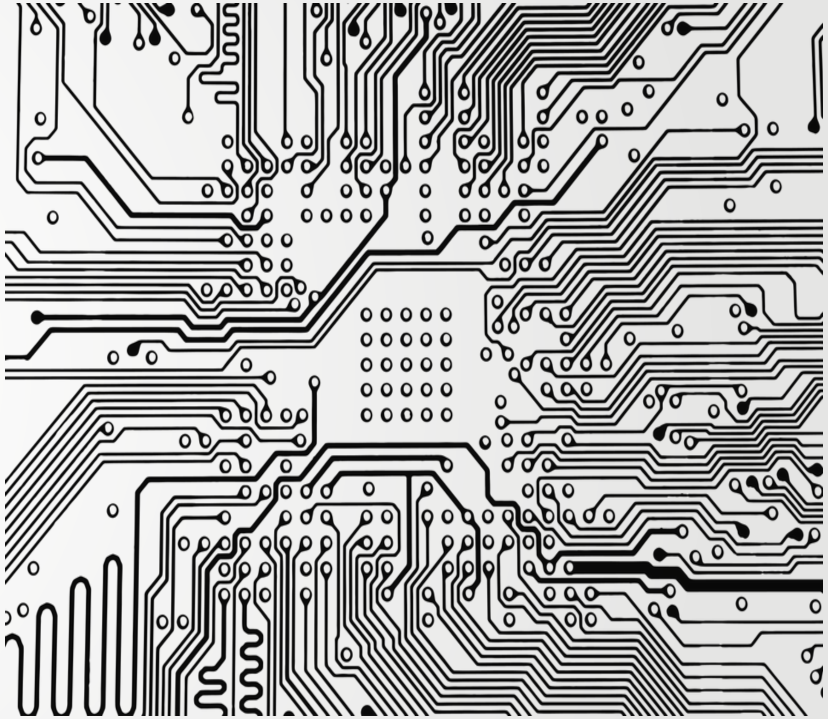
```
void teleBotSetup() {

  // Attempt to connect to Wifi network:
  Serial.print("Connecting Wifi: ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  client.setCACert(TELEGRAM_CERTIFICATE_ROOT); // Add root certificate for api.telegram.org

  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  bot.sendMessage(CHAT_ID, "Bot started up", "");
}
```

```
void alertUser(const String& message) {
  bot.sendMessage(CHAT_ID, message, "");
}
```



Forward Looking: PCB Design

- **Custom PCB will include:**
  - Atmega328p Microcontroller
  - ESP32 Wifi Microcontroller & Bluetooth
  - ESP32 WIFI Booster Antenna
  - HX711 Load Cell Amplifier
  - Adafruit MTK3339 - GPS
  - Adafruit 9 DOF LSM9DS1 - IMU
  - Dual TB6612FNG - Motor Driver
  - Voltage Regulator



BE REVOLUTIONARY™

COST ESTIMATE

University of Massachusetts Amherst

Cost Estimate

*Items sourced from M5 so actual cost is zero

Costs (Up to MDR)

Component	Predicted Cost
GPS	$39.06
Motor Driver	$5.45
Battery*	$12.99
Huzzah32 Feather*	$19.95
Proximity Sensor x4*	$15.80
Testing Platform	$29.99
Load Sensor x2	$15.86
IMU*	$14.95
Load Cell Amplifier	$6.05
Redboard*	$19.95
Photocells x3*	$2.85
Total	**$176.45**

Costs Estimate (MDR to CDR)

Component	Predicted Cost
Platform Materials	$80
Load Sensor x2	$15.86
PCB	$125
Antenna	$1.72
Atmega328p Chip	$2.86
ESP32 Chip	$3.15
Total	**$222.58**

Total Cost Estimate: $400.03

GPS - Adafruit MTK3339 B.O.B

- Location of system
- High-sensitivity receiver
- Low power

Photocell - CdS photoresistor

- Light sensor
- Light = ~1KΩ
- Dark = ~10KΩ

Ultrasonic Sensor - HC-SR04 x 4

- Very General Purpose
- Can measure distance

Lithium ion Battery Pack

- 2200mAh
- 11.1 V
- ~ 5 hours run time

IMU - Adafruit 9 DOF LSM9DS1

- Magnetometer, accelerometer, gyroscope
- Determine facing direction

Load Sensor x 4

- Cheap
- High Measurement range (up to 110kg or ~242.5 lbs)

Huzzah32 ESP32 Feather Board

- Wifi enabled
- Many GPIO pins
- Many ADC I/O pins
- SPI / I^{2}C (For GPS and Communication across boards/chips)

Redboard Qwiic - ATMega328p

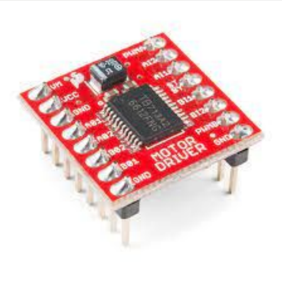
- General purpose MCU
- Cheap
- Analog & Digital IO
- Easy to prototype

ESP32 Antenna

- 2.4GHz Antenna
- 20 meters added distance

Load Cell Amplifier - HX711

- 24 bit precision
- Read the changes in the resistance



Motor Driver - Dual TB6612FNG

- Up to 15VDC
- H-Bridge for precise motor control

BE REVOLUTIONARY™

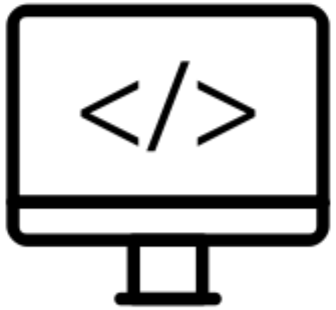
Responsibilities

University of Massachusetts Amherst

Team Member Responsibilities

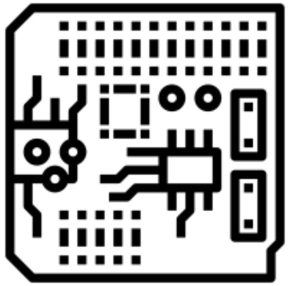
John Diep

- Software Lead
- Migration to Embedded C
- User Interface & Scheduler
- Systems Integration



Jasmine Hickey

- PCB Lead
- PCB Design
- IMU Integration
- Systems Integration



Smit Patel

- Budget Lead
- Motor Enclosure Scale-Up
- Antenna R&D
- GPS Navigation Design & Integration
- IMU Integration
- Systems Integration

Stephen Townsend

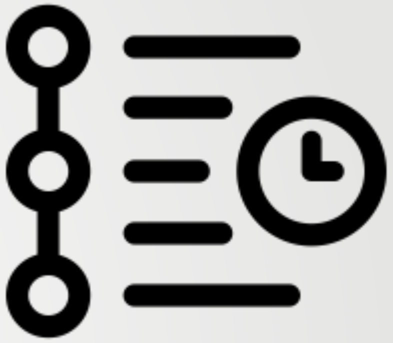
- Team Coordinator
- Motor Enclosure Scale Up
- GPS Navigation Design & Integration
- Systems Integration

BE REVOLUTIONARY™

MDR Deliverables

University of Massachusetts Amherst

Proposed MDR Deliverables (From-PDR)



- **Be able to preset a route for the system to navigate ✔**
- **Tabletop demonstration of Ultrasonic obstacle detection ✔**
    - Show 2 meter detection range specification
- **Tabletop demonstration of load detection ✔**
    - Show ≥ 5 lbs detection specification
- **Demonstrate a mobile system ✔**
- **Define platform for User Interface ✔**

Schedule Moving Forward

GANTT CHART (POST MDR -> CDR)								
		Finals Weeks (POST MDR)	Winter Break	Winter Break	Winter Break -> Spring 2022	Spring 2022	Spring 2022	Spring 2022 (Pre-CDR)
Task to be Done	Team Members	12/6 - 12/19	12/20 - 1/2	1/3 - 1/16	1/17 - 1/30	1/31 - 2/6	2/7 - 2/20	2/20 - 3/6
Large Scale Motor Enclosure	Stephen, Smit							
Antenna Design + Research	Smit							
PCB Design	Jasmine							
Migration to Embedded C	Stephen, John							
GPS Integration	Stephen, Smit							
IMU Integration	Smit, Jasmine							
Scheduler	John							
UI	John							

BE REVOLUTIONARY™

Demonstration

University of Massachusetts Amherst


QUESTIONS?
University of Massachusetts Amherst
BE REVOLUTIONARY™