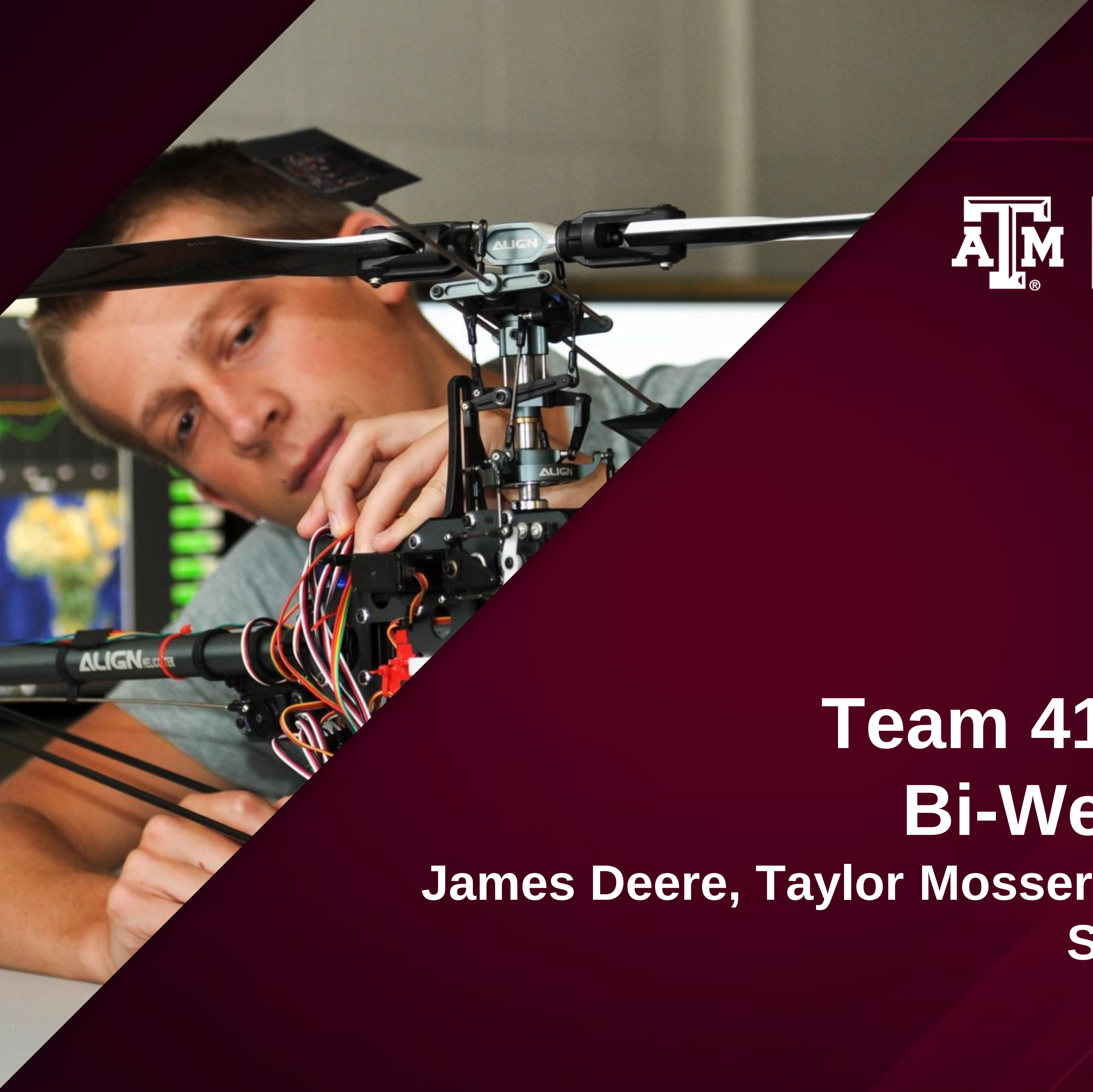


Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

Team 41: Roombotics Bi-Weekly Update 2

James Deere, Taylor Mosser, Todd Van Klaveren

Sponsor: Eric Robles

TA: Max Lesser

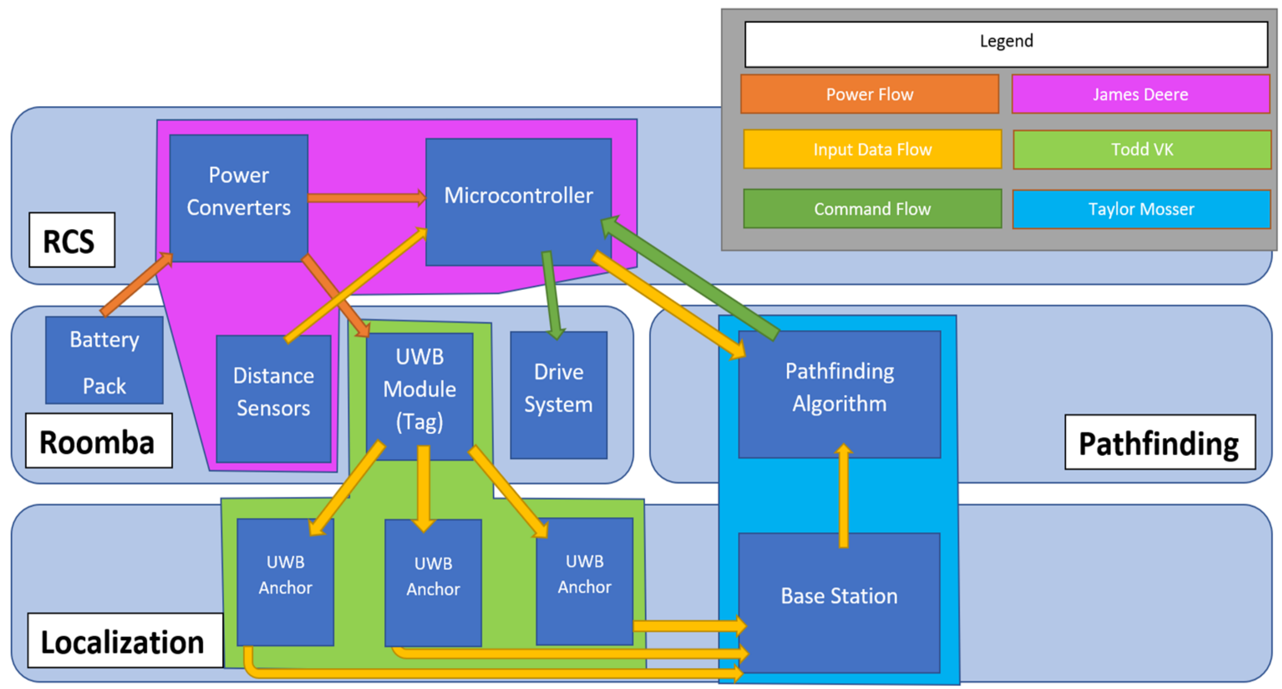
Roomba-MRS Project Summary

Problem: iRobot Roombas do not follow a specific path when cleaning a room and are prone to repeatedly cleaning the same areas or avoiding other areas altogether. While the Roombas can map the layout of a room, they do not have strategic methods of navigating the space to clean quickly and efficiently.

Additionally, one Roomba can be ineffective in cleaning larger spaces due to its small size. This system will allow for more than one Roomba to be deployed in a space in which they can work together to cover all area in the room.

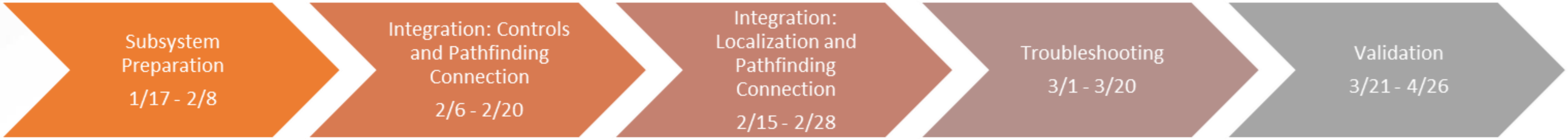
Solution: Roomba will navigate a space using the A* pathfinding algorithm to calculate lowest cost path from one point to another. Path calculation will be performed based on the location of the Roomba and any obstacles in the room. Roombas will not interfere with each other's assigned cleaning spaces and will avoid colliding with each other.

Project Overview



- Pathfinding Algorithm decides best path for each Roomba
- Base Station sends movement commands via Bluetooth
- Localization determines actual location of Roombas
- Feedback from localization updates Pathfinding algorithm.

Project Timeline



Subsystem Preparation
1/17 - 2/8

Integration: Controls and Pathfinding Connection
2/6 - 2/20

Integration: Localization and Pathfinding Connection
2/15 - 2/28

Troubleshooting
3/1 - 3/20

Validation
3/21 - 4/26

Subsystem preparation is complete, currently in integration between controls and pathfinding subsystems.

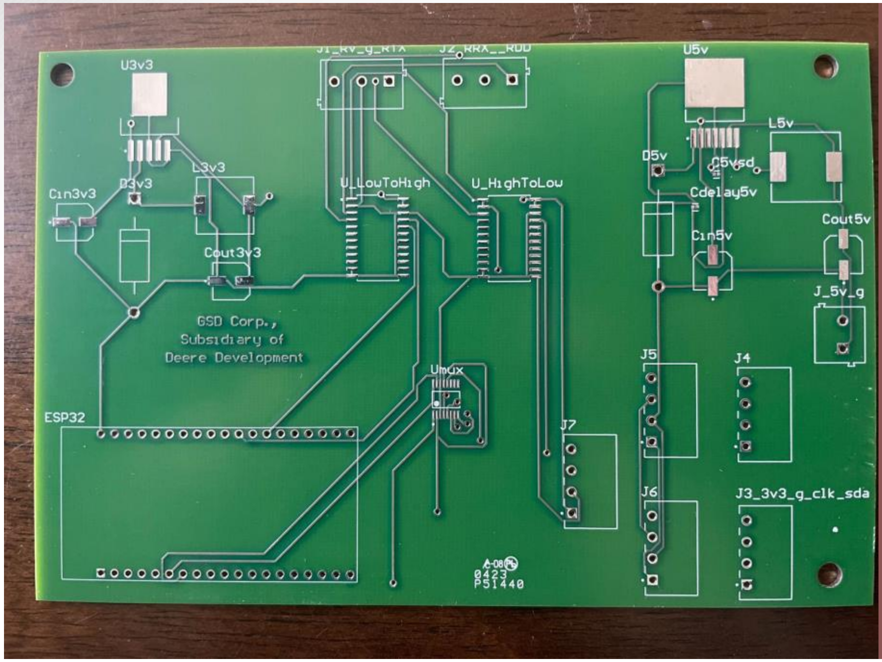
Roomba Control Subsystem

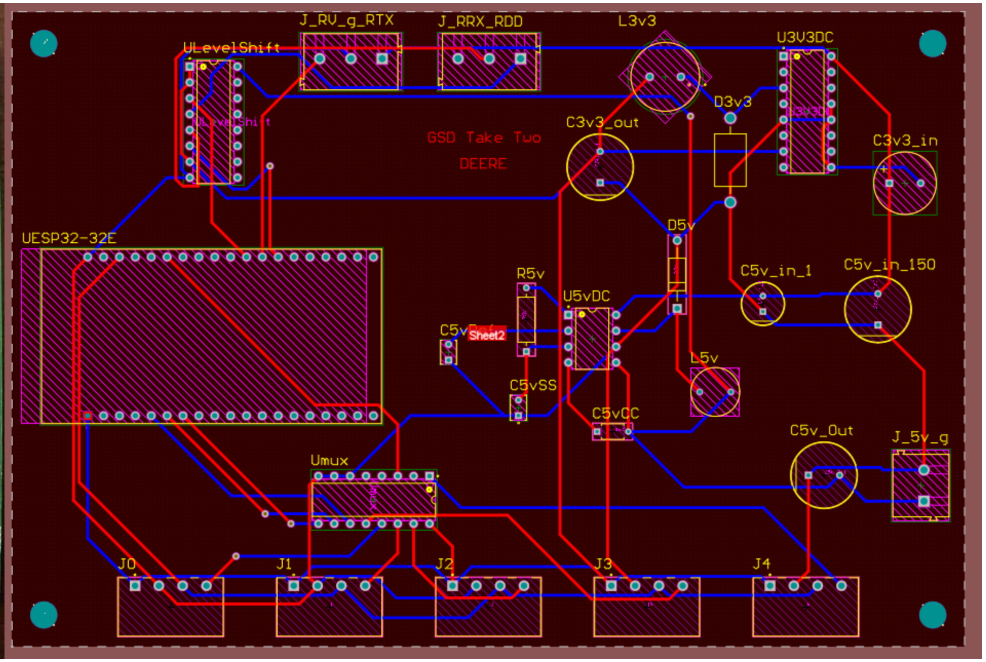
James Deere

Accomplishments since last update 25 hrs of effort	Ongoing progress/problems and plans until the next presentation
● PCB came in on night of 2/7 ● PCB soldered and tested on 2/8 - 2/10 (circuit failed test, probable short from soldering) ● New PCB designed with cheaper components that will be easier to test (trying to stay proactive on possible issues with PCB other than soldering short circuit). Order placed on 2/12, components already here. ● Tested switching multiplexer to support 5 Sensors	● Ongoing integration With pathfinding subsystem: have pathfinding subsystem connect to and control Roomba via bluetooth ● Additional soldered PCB to be tested in lab tonight (2/15) ● Ongoing learning of 3D CAD software to elegantly mount PCB, sensors, and localization module onto Roomba

Roomba Control Subsystem

James Deere





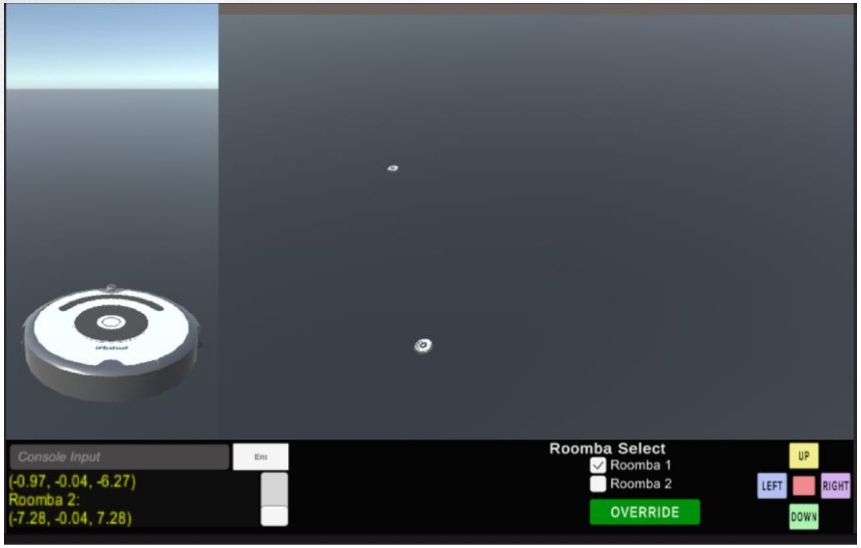
Pathfinding

Taylor Mosser

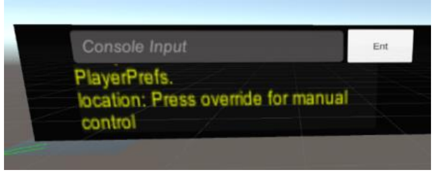
Accomplishments since last update 26 hours of effort	Ongoing progress/problems and plans until the next presentation
- Completed visual design for Unity UI - Console output takes commands from user input and displays information about the system - Pathfinding can be reinitialized and Roombas return to initial position - Coordinates of both Roombas can be found at any moment in simulation - Installed package for bluetooth connection between Unity and ESP32	- Finish adding commands to buttons in UI (on-click) - Run two scenes alongside each other: one for AI, other for representing real-life environment - Integrate pathfinding subsystem by connecting Unity environment using bluetooth library

Pathfinding

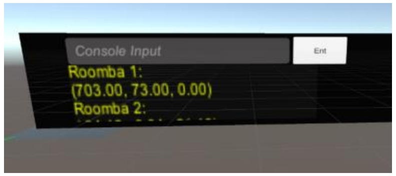
Taylor Mosser



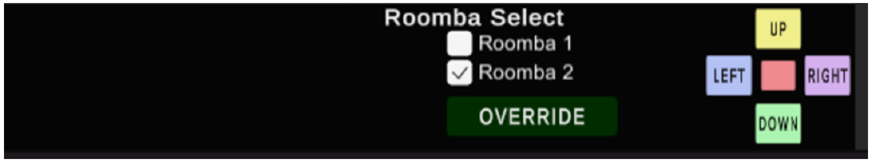
```
(-0.97, -0.04, -6.27)
Roomba 2:
(-7.28, -0.04, 7.28)
```

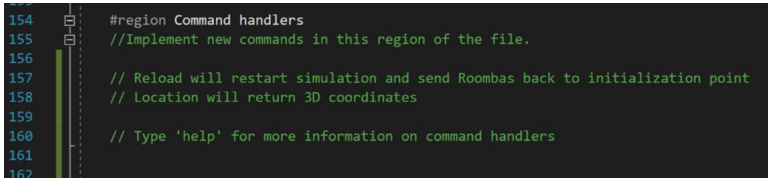


```
PlayerPrefs.
location: Press override for manual
control
```

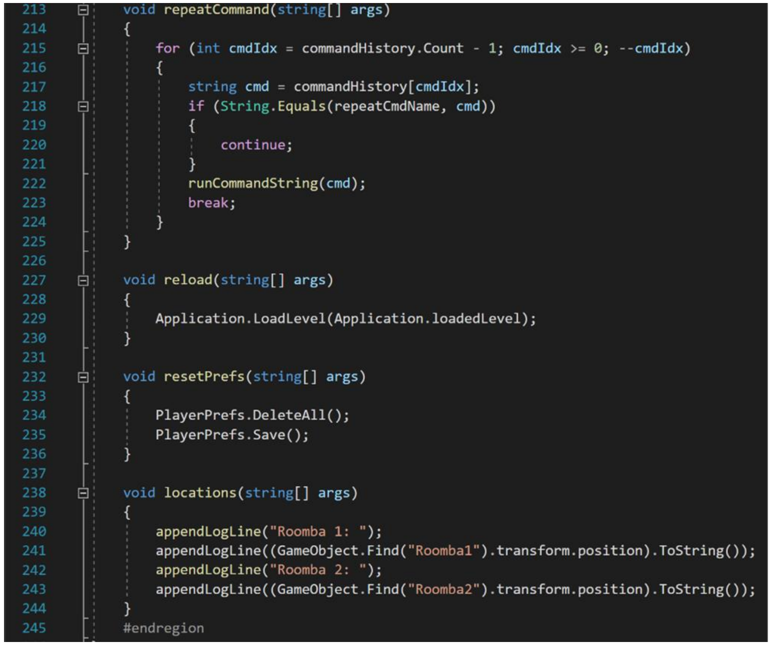


```
Roomba 1:
(703.00, 73.00, 0.00)
Roomba 2:
```





```
#region Command handlers
//Implement new commands in this region of the file.

// Reload will restart simulation and send Roombas back to initialization point
// Location will return 3D coordinates

// Type 'help' for more information on command handlers

```



```
void repeatCommand(string[] args)
{
    for (int cmdIdx = commandHistory.Count - 1; cmdIdx >= 0; --cmdIdx)
    {
        string cmd = commandHistory[cmdIdx];
        if (String.Equals(repeatCmdName, cmd))
        {
            continue;
        }
        runCommandString(cmd);
        break;
    }
}

void reload(string[] args)
{
    Application.LoadLevel(Application.loadedLevel);
}

void resetPrefs(string[] args)
{
    PlayerPrefs.DeleteAll();
    PlayerPrefs.Save();
}

void locations(string[] args)
{
    appendLogLine("Roomba 1: ");
    appendLogLine((GameObject.Find("Roomba1").transform.position).ToString());
    appendLogLine("Roomba 2: ");
    appendLogLine((GameObject.Find("Roomba2").transform.position).ToString());
}
#endregion
```

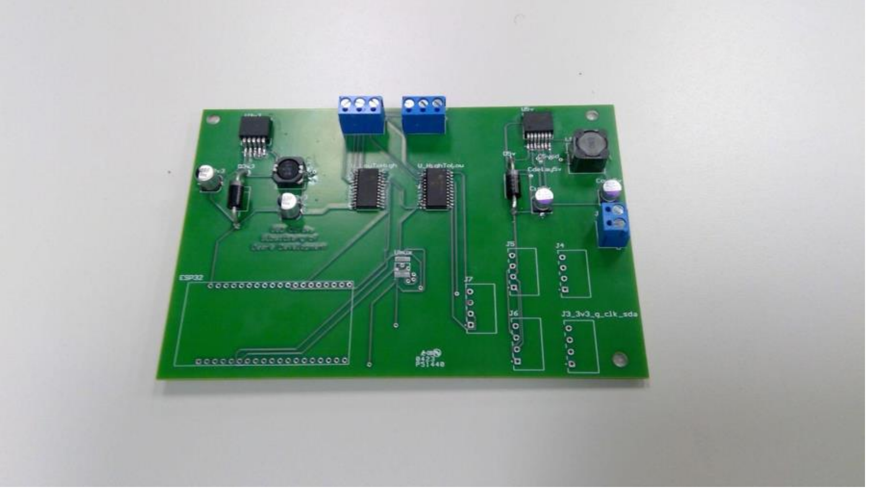
Localization Subsystem

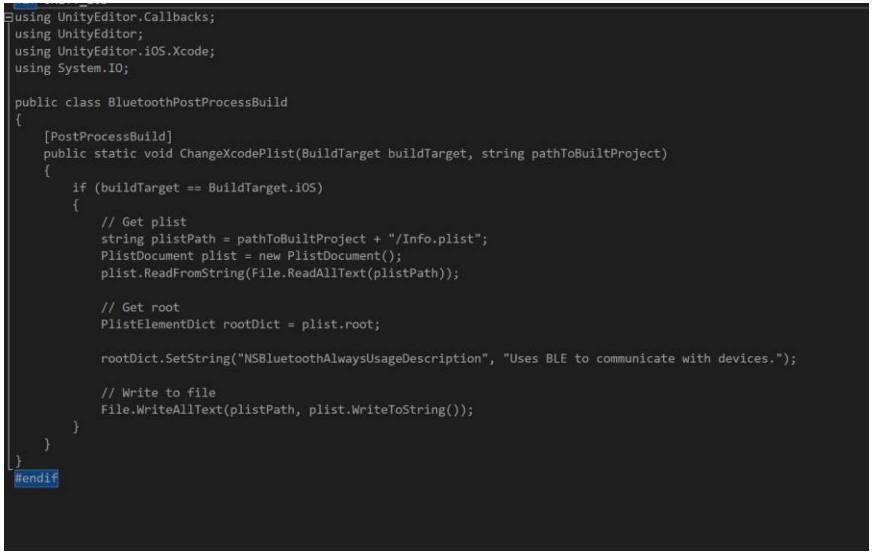
Todd Van Klaveren

Accomplishments since last update 20 hrs	Ongoing progress/problems and plans until the next presentation
No change to localization subsystem. Helped James to solder first board. Board failed initial testing. Have bluetooth library loaded in unity.	Currently fixing first board and testing again after fixes. Working on program making BT connection, and receiving commands. Plan to have board soldered/tested in lab today, and BT integrated fully by Monday.

Localization Subsystem

Todd Van Klaveren



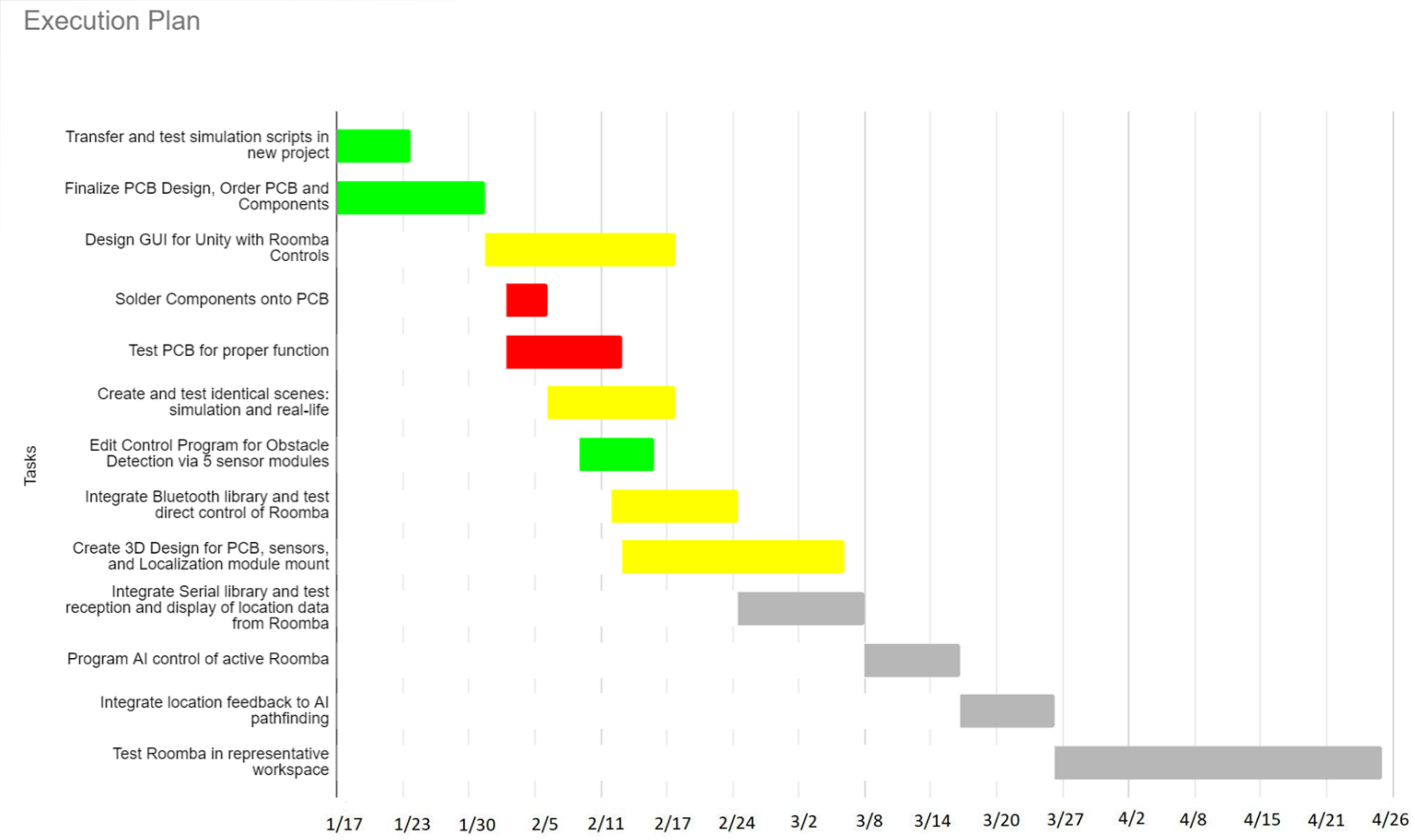


```
using UnityEditor.Callbacks;
using UnityEditor;
using UnityEditor.iOS.Xcode;
using System.IO;

public class BluetoothPostProcessBuild
{
    [PostProcessBuild]
    public static void ChangeXcodePlist(BuildTarget buildTarget, string pathToBuiltProject)
    {
        if (buildTarget == BuildTarget.iOS)
        {
            // Get plist
            string plistPath = pathToBuiltProject + "/Info.plist";
            PlistDocument plist = new PlistDocument();
            plist.ReadFromString(File.ReadAllText(plistPath));

            // Get root
            PlistElementDict rootDict = plist.root;

            rootDict.SetString("NSBluetoothAlwaysUsageDescription", "Uses BLE to communicate with devices.");

            // Write to file
            File.WriteAllText(plistPath, plist.WriteToString());
        }
    }
}
#endif
```

Parts Ordering Status

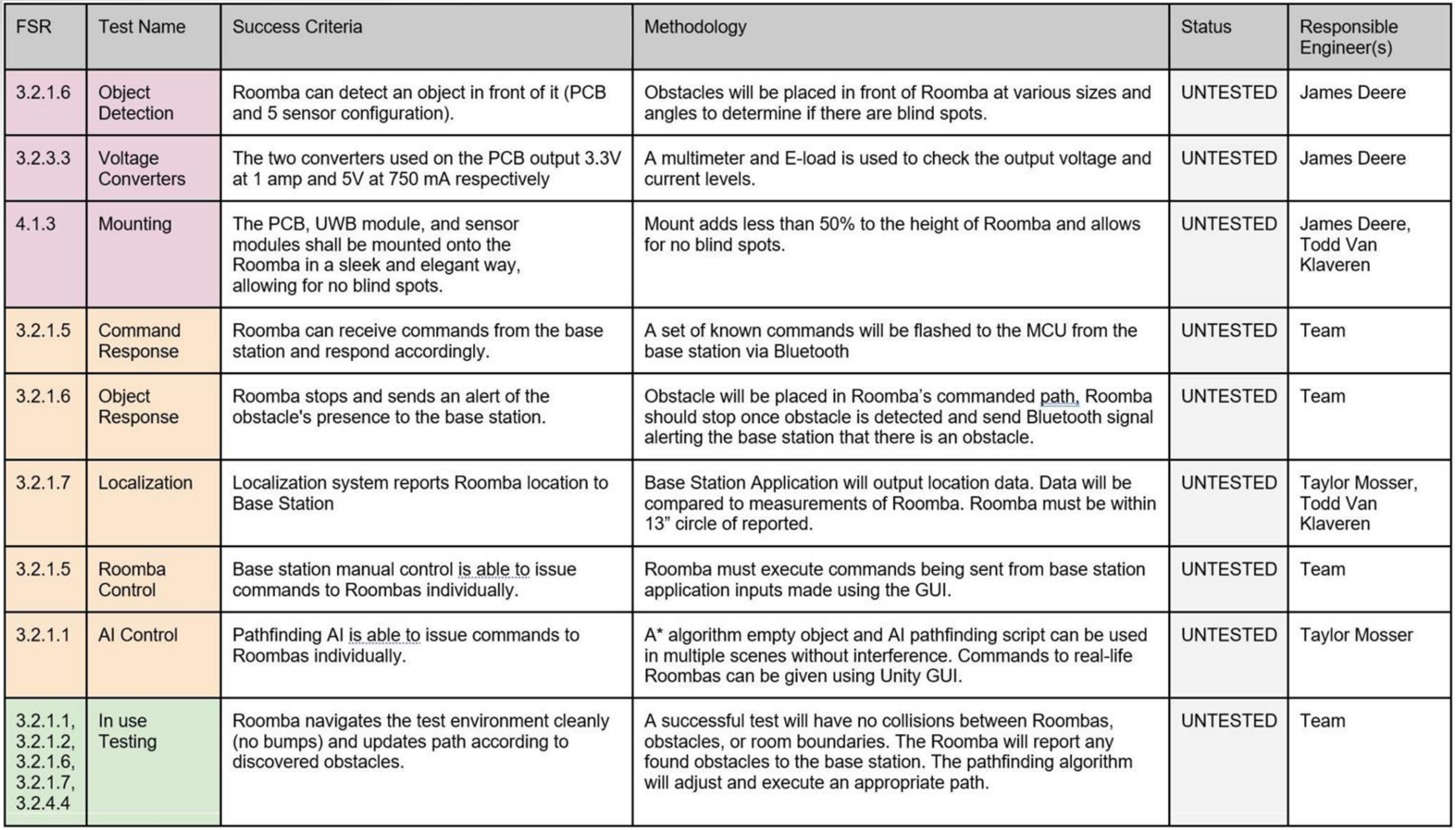
- New PCB ordered on 2/12, shipped on 2/15
- Components for new PCB have already arrived

Execution Plan



🔴 Behind Schedule	🟢 Completed	🟡 In Progress	⚪ Not Started

Validation Plan



FSR	Test Name	Success Criteria	Methodology	Status	Responsible Engineer(s)
3.2.1.6	Object Detection	Roomba can detect an object in front of it (PCB and 5 sensor configuration).	Obstacles will be placed in front of Roomba at various sizes and angles to determine if there are blind spots.	UNTESTED	James Deere
3.2.3.3	Voltage Converters	The two converters used on the PCB output 3.3V at 1 amp and 5V at 750 mA respectively	A multimeter and E-load is used to check the output voltage and current levels.	UNTESTED	James Deere
4.1.3	Mounting	The PCB, UWB module, and sensor modules shall be mounted onto the Roomba in a sleek and elegant way, allowing for no blind spots.	Mount adds less than 50% to the height of Roomba and allows for no blind spots.	UNTESTED	James Deere, Todd Van Klaveren
3.2.1.5	Command Response	Roomba can receive commands from the base station and respond accordingly.	A set of known commands will be flashed to the MCU from the base station via Bluetooth	UNTESTED	Team
3.2.1.6	Object Response	Roomba stops and sends an alert of the obstacle's presence to the base station.	Obstacle will be placed in Roomba's commanded path, Roomba should stop once obstacle is detected and send Bluetooth signal alerting the base station that there is an obstacle.	UNTESTED	Team
3.2.1.7	Localization	Localization system reports Roomba location to Base Station	Base Station Application will output location data. Data will be compared to measurements of Roomba. Roomba must be within 13" circle of reported.	UNTESTED	Taylor Mosser, Todd Van Klaveren
3.2.1.5	Roomba Control	Base station manual control is able to issue commands to Roombas individually.	Roomba must execute commands being sent from base station application inputs made using the GUI.	UNTESTED	Team
3.2.1.1	AI Control	Pathfinding AI is able to issue commands to Roombas individually.	A* algorithm empty object and AI pathfinding script can be used in multiple scenes without interference. Commands to real-life Roombas can be given using Unity GUI.	UNTESTED	Taylor Mosser
3.2.1.1, 3.2.1.2, 3.2.1.6, 3.2.1.7, 3.2.4.4	In use Testing	Roomba navigates the test environment cleanly (no bumps) and updates path according to discovered obstacles.	A successful test will have no collisions between Roombas, obstacles, or room boundaries. The Roomba will report any found obstacles to the base station. The pathfinding algorithm will adjust and execute an appropriate path.	UNTESTED	Team

Questions?

Thank you!