AGV Written Project Report

AR West Point Fortress

Daniel An

Dillon Bliss

Austin Pearson

Haley Steele

Table of Contents

1.0 Problem Definition

1.1 Introduction

Robots have become extremely important for military operations in recent years. The ability to enter a room or hostile environment and perform tasks without risking soldiers is very desirable. In order to protect soldiers we need robots that can perform given tasks. This includes autonomous movement around objects, sensing lights and objects in the area, and many more capabilities. Our project is to take our robot, “Cynthia,” and make it as autonomous as possible to see what possible capabilities robots could utilize in a combat environment.

1.2 Problem Statement/Product Vision

The ARWP team needs to build an autonomous vehicle that can navigate through a given course in less than 5 minutes. The obstacles will be bounded by both real and virtual walls (virtual wall depicted by black tapes). The robot must navigate within the area to locate an opened and darken room, enter the room, and navigate towards a light source within the room.

2.0 Requirements Analysis Phase

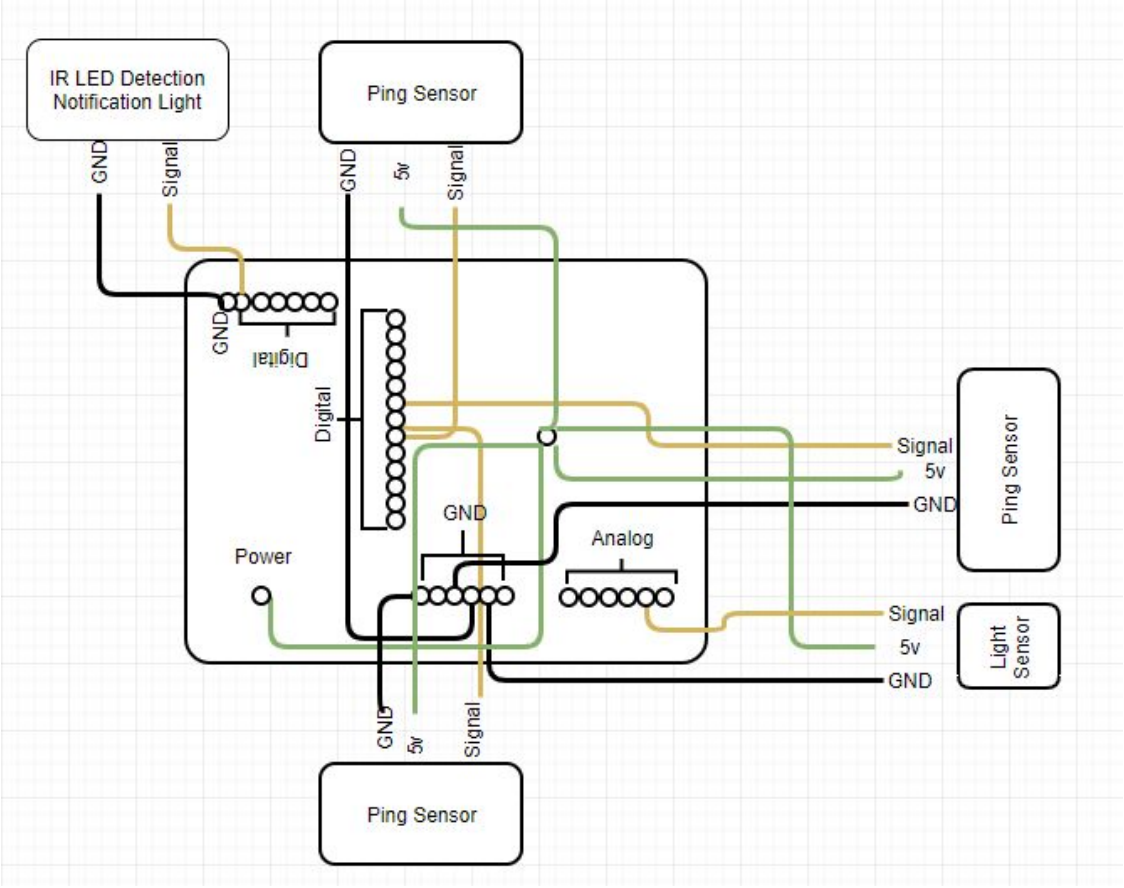
2.1 Specified and Implied requirements

There are many specified and implied requirements that are included in this project. Listed below are the requirements that were given to us in the initial project guidance as well as implied ones that we discovered throughout our design phase.

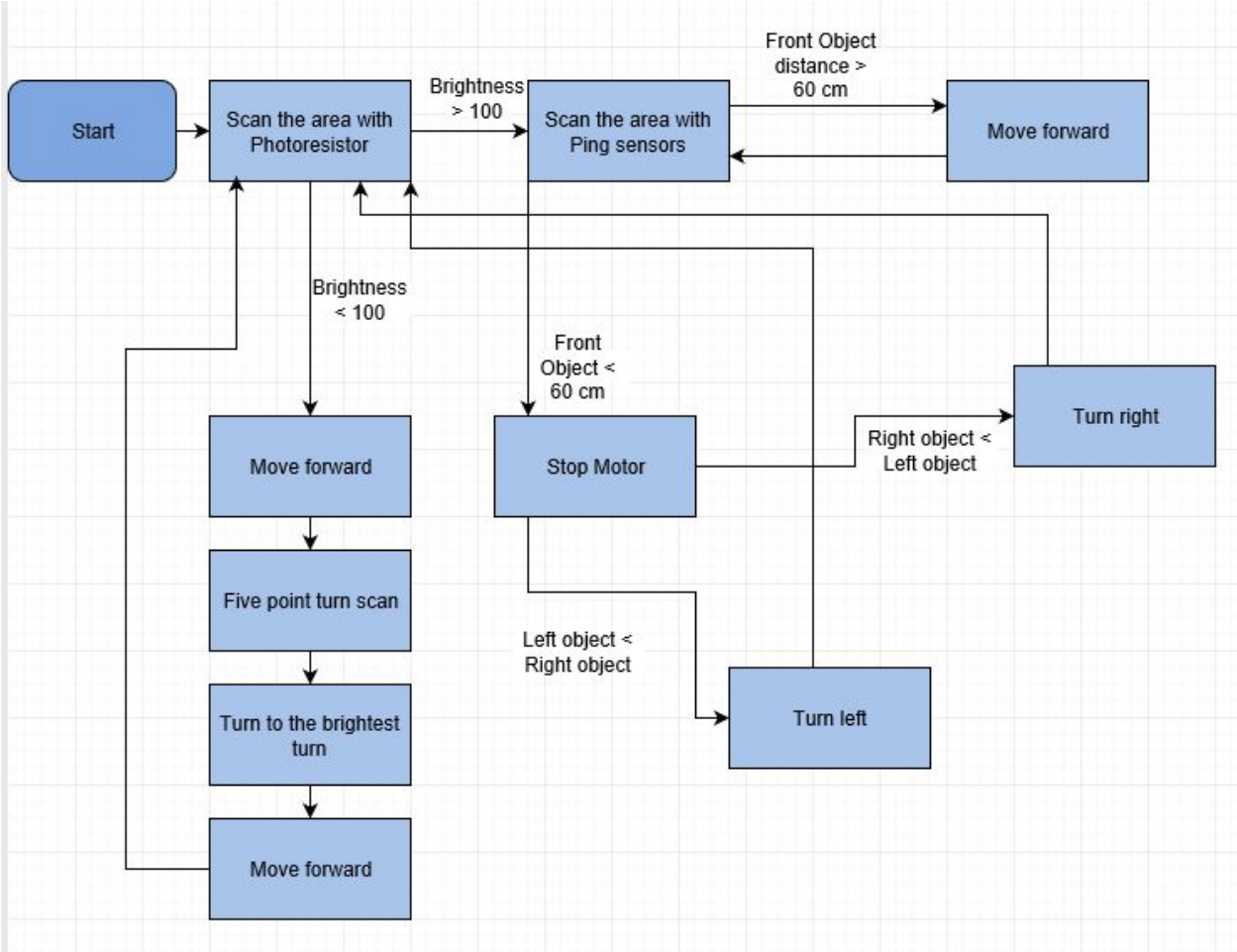
- Specified Requirements:
    - Course completed in 5 minutes or less
    - Built from a MMP-5 Mobile Robot Platform
    - Using Parallax BoE with the Basic Stamp 2 microcontroller OR Arduino microcontroller board with Parallax BoE shield
- Implied Requirements:
    - Use Sensors to help with the navigation of the course
    - Minimal to no touching the AGV once it starts the course

3.0 Systems Design Phase

3.1 Functional Block Diagram



3.2 Flow Chart



3.3 Design Methodology

Our first task was to make sure our robot could move in a straight line and stop on its own. From there we added in rotation capability. Writing programs that would make the robot turn 90 degrees in either direction as well as 180 degree turns. We then added on ping sensors that would allow the robot to detect objects from the front, right, and left. We implemented

algorithms that would integrate the ping sensors and turning functions to allow the robot to navigate obstacles that it might come across on its own.

4.0 Detailed Design, System Integration & Test Phase

4.1 Subsystem Design and Tests

We started our subsystem testing with testing the servo motors. First, we took the code from the website and modified it to have four different functions; forward, backwards, left turn 90 degrees and right turn 90 degrees. We tested these functions on the tile floor by using the lines to ensure it was traveling in a straight line and making full 90 degree turns.

Next, we tested the ping sensors. We set up code that outputted the distances detected by the sensor and we used a yard stick and piece of paper to insure the readings were accurate.

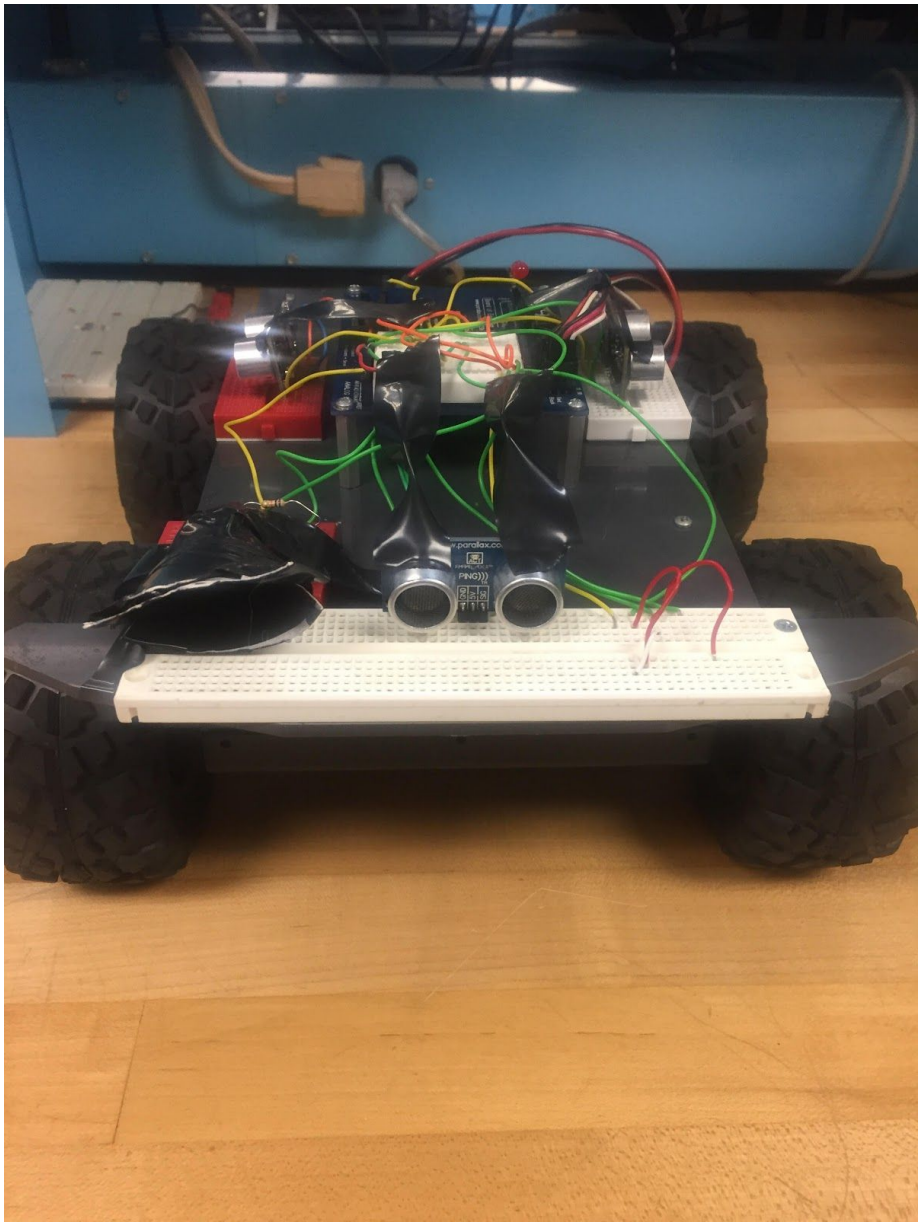
Lastly, we added on the photo sensor. We tested it by running code that outputted the brightness level it detected to the serial monitor. We tested it in different rooms to determine what level of brightness it would detect in a dark room with a single light. We then added a cone to make the light sensor directional and tested it at different directions to insure it would detecting the different levels of brightness in a dark room.

4.2 Integration Tests and Results

We integrated the ping sensors together as well as with the servo motors. Determining the distance we wanted the robot to stop from an object. We tested this by running the robot towards an object and analyzing the distances we determined would be best fit for the actual run. We determined that 75 cm would be the right distance for the front sensor to detect objects which would stop the robot at roughly 30 cm from the object. We then added a second and third sensor on the right and left side by performing the same process we used to put on the the first sensor. We then added logical statements to the code that would provide the robot with a list of rules for when the robot comes into contact with an object at the front and side. We tested the side distances until we determined that a distance around 50 cm would be sufficient.

5.0 Properly Functioning System Phase

5.1 Final Results



Our AGV was able to successfully navigate rudimentary obstacles when at full battery charge. A rudimentary obstacle can most nearly be described as a solid object with a large enough mass that would allow the ping sensor to detect its presence. Successful navigation includes the detection of the obstacle using the front ping sensor and demanding a response from the servo-motors that would turn the servo-motors left or right depending on the distance picked up from the two ping sensors on either side of the AGV. However, the space in the obstacle courses were smaller than anticipated; thus, our ping sensors did not work properly in the narrow hallway. Additionally, our light sensor was able to function properly under controlled lab conditions. Controlled lab conditions can most nearly be described as operating in a large dark room with a single source of light without light pollution from the hallway that would interfere with the sensor. When operating under controlled lab conditions, our AGV successfully navigated towards the brightest light source in the room. This was achieved by passing the input received by the light sensor into our code algorithm which utilized the servo-motors to swivel the AGV and collect input data by the light sensor at the outermost edges of the AGV's swivel motion. Once the data was collected and stored in variables to be used for our algorithm, a simple conditional statement was used to determine which direction was delivering the greatest amount of light to the light sensor.

During our trial runs, our AGV was able to detect rudimentary obstacles, however, the elicited response from the servo-motors was contrary to our lab tests. Similarly, our AGV prematurely detected the dark room from the shadows of bystanders observing the course. Once inside the dark room, our AGV was picking up light pollution from the hallway as the nearest source of bright light.

5.2 Conclusion

In conclusion, our group learned a lot about autonomous ground vehicle and also a lot about each other. This project has set the precedence for our future group projects, and we will utilize this experience to build better team cohesion and effectiveness. In terms of AGV, we learned about how sensors can project algorithms onto the physical world. As computer scientists, we never experienced our algorithm being used in the physical world. However, using Arduino and some hardware, we can create a vehicle that can potentially be completely autonomous. This project was an interesting mix that challenged the CS/IT majors, and we hope to work on something like this in the future.

Appendices

Appendix 1 References

https://www.arduino.cc/en/Tutorial/Ping
(Ping Sensors arduino code)

https://learn.adafruit.com/photocells/using-a-photocell
(Photoresistors arduino code)

Appendix 2 URL for Group GitHub

https://github.com/hsteeleIO/AVG