Tyler Thompson

1/23/23

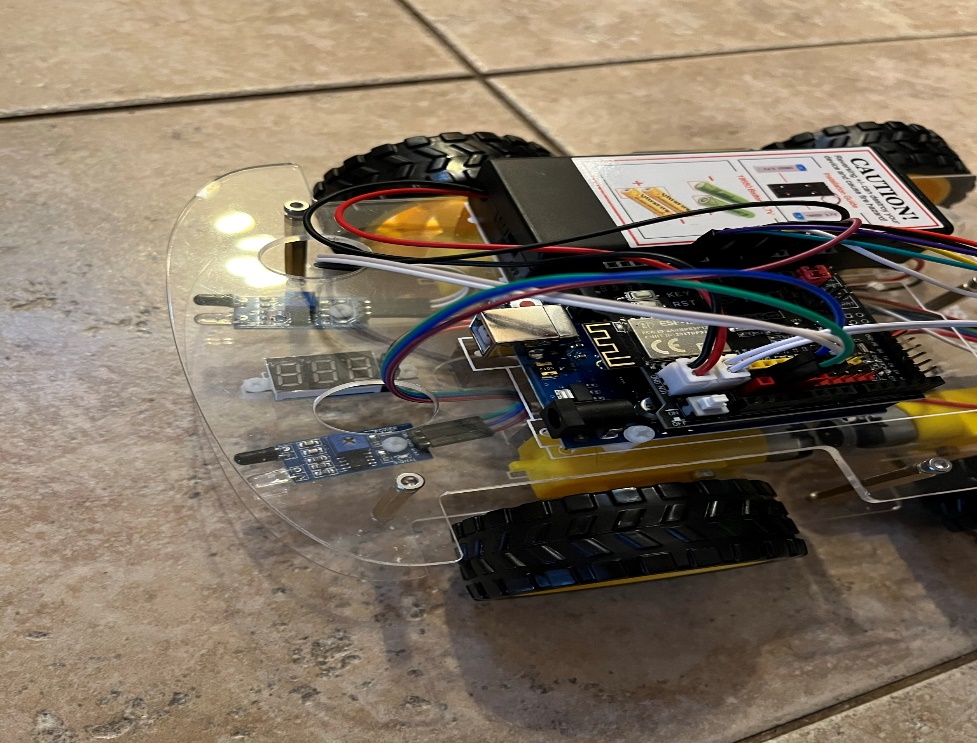
tjt5498@psu.edu

**Lab 5**

**Objective**

Have the robot avoid objects and not allow it to run into anything. Set up an obstacle course to test the program. Make robot move in a direction if it is needed and back up if there is nowhere for the robot to continue moving forward. The robot will move forward until something is encountered.

**Robot With Sensors**



**Pseudocode**

This is how the main function within the program operates. It is used within a loop so it can continue for however long is needed.

void avoid\_Object(){

  Start sensors

 if (Both sensors are 0) // If something is in front.

 {

    Stop

Move Back

Turn Right

 }

 else if (Both sensors are 1) // Moves the robot in the direction of the sensors.

 {

     go\_Back // Go forward.

    }

 else if (Left sensor is on) // Turns the robot away from object.

 {

  //only left sensor detect obstacle

      back\_Right // Turn right

 }

  else if (Right sensor is on) // Turns the robot away from object.

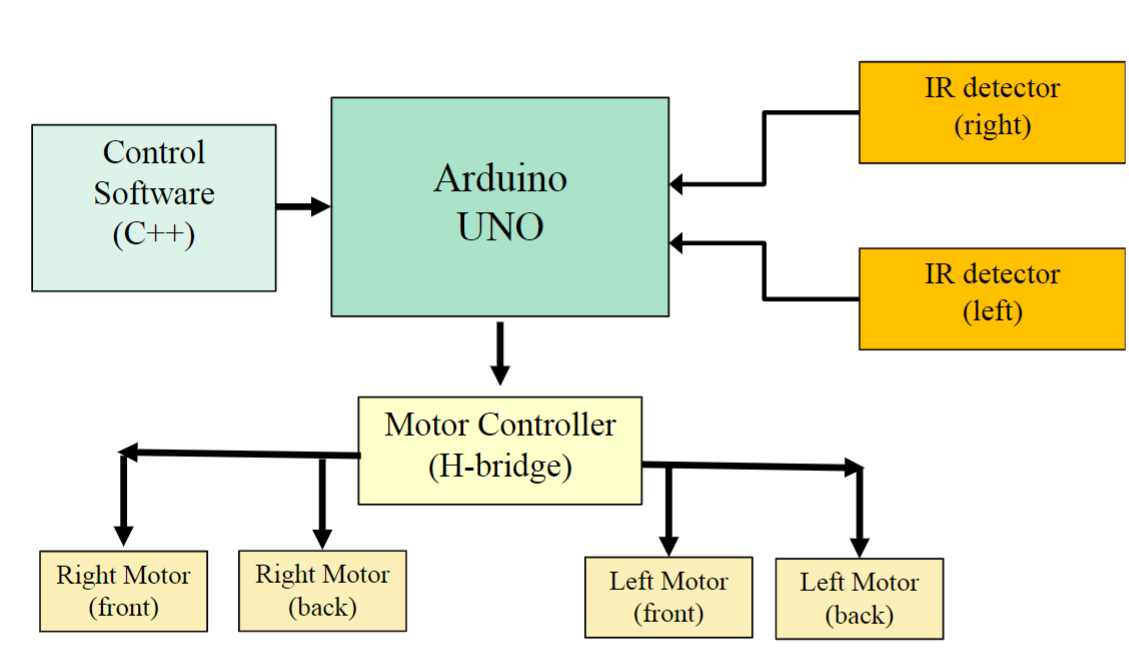
 {

   //only right sensor detect obstacle

      back\_Left // Turn left

}

}

**System Block Diagram**

**Results**

|  |  |
| --- | --- |
| Robot Moving Through Obstacle Course  <https://share.icloud.com/photos/0deXhKJAvP1bJa-dr98efh49w> | Here we can see the robot successfully using all motions necessary to avoid running into an object. The correct sensor is activated if the robot needs to move a certain direction. Sensors were adjusted to make sure proper range was correct. It was determined through testing that 6 inches proved to be the best distance for the robot to detect and object. |
| Robot Moving Away From Right  <https://share.icloud.com/photos/021TSCJkfmqoWYGHm8-Unqqlw> | Sensor correctly moving the robot away from the right object. It was determined that 6 inches was the best distance for the robot to properly detect an object. |
| Robot Moving Away From Left  <https://share.icloud.com/photos/0d7KF_fkb6cA8kXumfaO9OTyg> | Sensor correctly moving the robot away from the left object. It was determined that 6 inches was the best distance for the robot to properly detect and object. |
| Robot Moving Back From Object  <https://share.icloud.com/photos/0a76n6hzycJ0YAUCk1nqrBKIw> | Sensors properly detecting something in front of the robot. This then causes the robot to reverse and spin for a certain amount of time to ensure that it moves in a different direction. This test in particular showed why it was important to adjust the sensors. With the sensors set at 6 inches the robot could find a new suitable path properly. |

**Code**

The rest of the code can be found within the project file. Here is the function that is called within a loop to make sure the robot avoids objects.

// Make the robot avoid objects. Starts after a 1s delay.

void avoid\_Object(){

  int IRvalueLeft= digitalRead(RightObstacleSensor);

  int IRvalueRight=digitalRead(LeftObstacleSensor);

 if (IRvalueLeft==LOW && IRvalueRight==LOW) // If something is in front.

 {

    stop\_Stop();

    delay(1000);

    go\_Advance();

    delay(1500);

    back\_Right();

    delay(1000);

 }

 else  if (IRvalueLeft==HIGH && IRvalueRight==HIGH) // Moves the robot in the direction of the sensors.

 {

     go\_Back();

    }

 else if (IRvalueLeft==LOW && IRvalueRight==HIGH) // Turns the robot away from object.

 {

  //only left sensor detect obstacle

      back\_Right(); // Turn right

 }

  else if (IRvalueLeft==HIGH && IRvalueRight==LOW) // Turns the robot away from object.

 {

   //only right sensor detect obstacle

      back\_Left(); // Turn left

}

}

**Conclusion**

The program was a success when programmed into the robot. There are a few major take aways from this lab. The sensors for the robot need to be adjusted thoroughly to ensure that they can see an object around the same distance. Also speed is a factor since the robot does not want to be moving fast or else the sensors might not detect an object in time. It is good to use an overall speed and make slight adjustments when needed. A lot of testing was needed to determine the optimal distance to detect and object. It was determined that 4 inches did not provide the robot with enough time to stop without possibly hitting an object. After a few initial test 6 inches proved to yield the best possible results for this program. In addition to all that, when installing new components into the robot it is useful to remove the top part to ensure a smooth and proper installation. The robot was able to move around the obstacle course with ease and perform the necessary moves required.