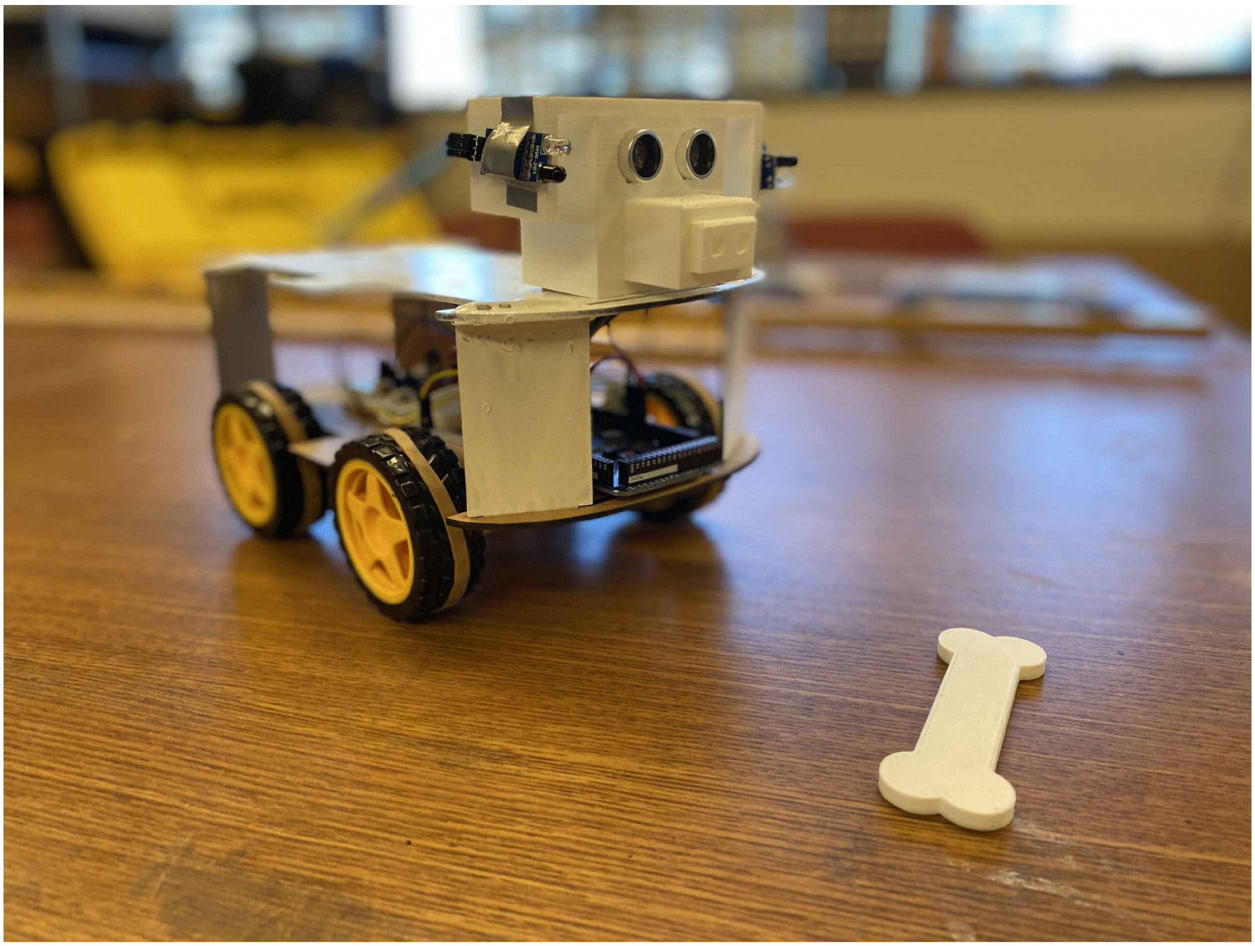
Rakshit Dabhi
Rahul Aggarwal

Treat-Following Dog Robot



Project Introduction

As the title of our project suggests, we created a treat-following dog robot, which uses an ultrasonic sensor and two IR obstacle-detection sensors as inputs to drive its motors towards a hand with a treat in it. To make the robot look like a dog, we hid the complex wiring and circuitry inside of the chassis, 3D printed a dog head to fit the sensors, and painted it to resemble a white dog.

Hardware and Other Physical Parts

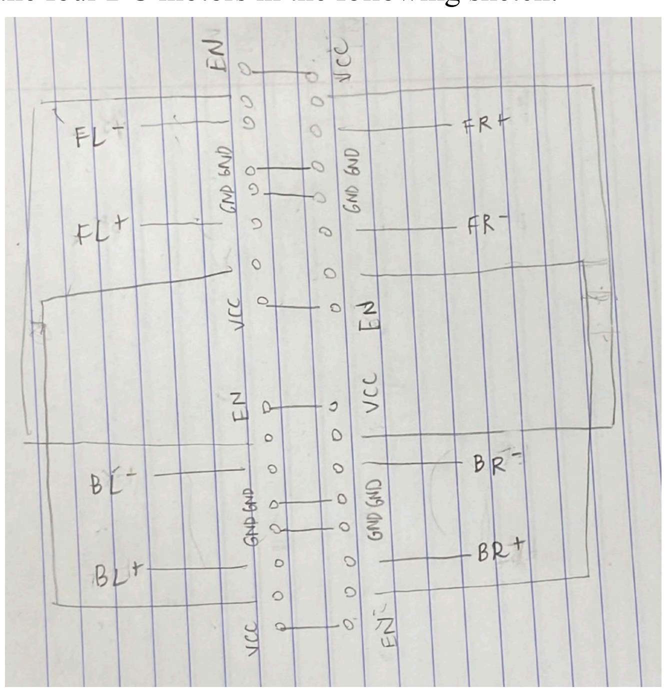
For the project, we used an ultrasonic sensor, two IR obstacle-detection sensors, an Arduino Mega, a 9V supply battery pack, connection wires, a breadboard, two L298n dual H-bridge drivers, four DC motors, four rubber wheels with added rubber bands for enhanced grip, a laser-cut chassis, a 3D-printed dog head and dog bone, a glue gun, and some white spray paint.

Custom Designed by Us

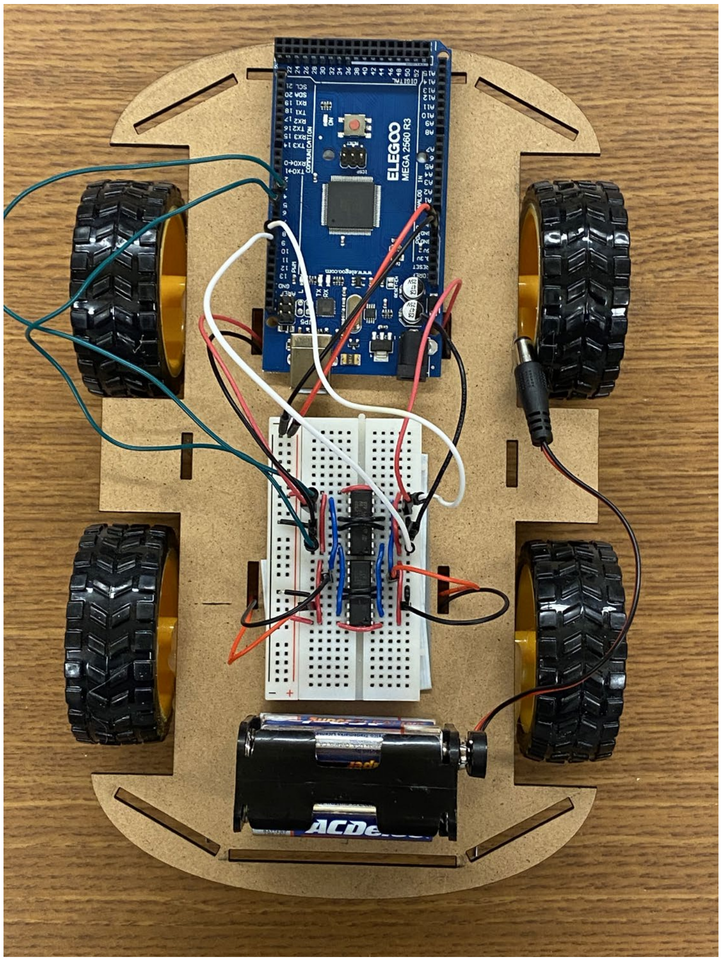
For this project, we laser-cut the chassis parts and 3D-printed a dog head and dog bone by ourselves. This gave us the flexibility to design the robot for our purposes and make it look more like a dog. The laser-cut chassis had customized holes to pull motor wires through and attach both laser-cut base plates to each other through laser-cut bars. The dog head also had a huge hole on the bottom and the back side to pull wires through to connect to the sensors. In addition, the dog head had eye-socket holes specifically designed with the dimensions of the ultrasonic sensor to put through to make it look like dog eyes.

Circuitry and Wiring

To create the logic for the motor drive, we first planned out the L298n dial H-bridge driver pin inputs and outputs for the four DC motors in the following sketch:



Using this diagram above, we were able to produce the following circuit:



Algorithm

For the code, we didn't use any libraries and everything was built by us. The general algorithm for how this dog robot works is that it first reads the distance reading from the ultrasonic sensor in centimeters. It then reads a binary value reading from both the left and right IR sensors and inverts it, where a value of 0 indicates that there is no obstacle, and a value of 1 indicates that there is an obstacle in front of the sensor. This lets the robot know whether it needs to turn left or right depending on the values because a moving hand will trigger these sensors and indicate the direction the robot needs to turn. Once it has these three sensor values, the robot turns left if the left IR sensor is triggered and the right IR sensor is not triggered, turns right if the left IR sensor is not triggered and the right IR sensor is triggered, moves forward if both IR sensors read the same value and the ultrasonic sensor distance reading is within the range of 12cm to 30cm, moves backward if both IR sensors read the same value and the ultrasonic sensor distance reading is less than 7cm, else it stops. You can see the full code in the ZIP archive attached with this file inside the folder named "Treat_Following_Dog_Robot_Code".

Implementation Process

1. We first laser-cut and built the chassis and glued the motors and wheels to create a four-wheeled robot
2. We tested the robot's range of motion by having it go forward, backward, and rotate left and right
    a. From these calibration tests, we were able to figure out that instead of having the robot turn in-place by having one set of wheels spin clockwise and the other set of wheels turn counterclockwise, we needed to have only one set of wheels turning and the other side not turning at all
3. We then tested each individual sensor (ultrasonic sensor and left and right IR detection sensors) to figure out how to get readings and understand the physical range of these sensors
    a. We learned that the IR sensors could only accurately detect an obstacle at a maximum of 30cm away, so this was the range limit we set for our robot dog
4. We then integrated the ultrasonic sensor and IR sensors into the robot and wrote some code to take these sensor values and move the robot accordingly (algorithm is described in a dedicated section called "Algorithm")
5. Once the robot was working as expected, we spray-painted the shell of the robot to a white color to make it look more appealing
6. We then 3D printed both the dog-head and dog-bone in a white plastic filament to match the color of the shell and give the robot the appearance of a white dog
7. We attached the sensors to the dog head and integrated this head to the robot
8. We again tested the code and found that our IR sensors were false-triggering all of a sudden
    a. After switching out the IR sensors multiple times and placing them at different positions on the dog head, we were finally able to get the IR sensors to stop false-triggering
    b. Turns out that at a particular location inside the lab, IR radiation was being emitted/reflected into the IR receiver which caused the false-triggering
9. After testing again, we realized that the batteries were draining very quickly, and the battery pack was too heavy, causing the center-of-mass of the robot to shift backwards, so we decided for demonstration purposes that we will change the location of the battery pack to be on top of the dog
10. Finally, the robot dog worked as intended!!!

Skills Learned

One of the most important skills we learned was how to work in a group setting to develop a prototype from the ground up. Through this work-setting, we learned how to trust our partner's work and hold each other accountable for our work, as well as cultivate a healthy environment for debates and arguments about different implementation methods. Beyond soft skills, we also learned how to work with ultrasonic sensors and take distance measurements in centimeters, and we learned how the IR obstacle-detection sensors worked and how to read their input values. We were also able to gain more insight into design problems such as weight distribution, motor power output, and battery requirements based on consumption and current (Amps) demands. Lastly, we learned how to make a prototype device look cleaner, aesthetically pleasing, and "production-ready".

Challenges

One of the biggest challenges that we faced when creating this robot was that two of the DC motors weren't delivering as much power as the other two DC motors, so the turning radius in one direction was a lot larger than the turning radius in the opposite direction. We also had challenges with weight distribution. Originally, we intended to have the battery pack and the Arduino Mega in the middle to evenly distribute the weight, but due to physical constraints with the length of the motor wires, we had to reorganize the circuitry which forced us to push the battery pack all the way to the backside. As a result, the robot was slightly tilted backwards, and the heavier weight in the back put a lot more force on the back two motors while the front two motors barely touched the ground. The first problem mentioned prior (two DC motors were weaker than the other two DC motors) exasperated this problem because when the robot was turning, the turns would be very slow because the back motor was essentially the only motor responsible for turning. To work around this problem, we shifted the battery pack's location from inside the dog to the top-middle part of the dog to give a more even weight distribution, and added motor assistance from the other inactive motors so that they wouldn't produce drag/friction force on the two motors that were turning the dog robot.

Work Distribution

<table>
<tr><th>Team Member Contribution</th><th>Rakshit Dabhi</th><th>Rahul Aggarwal</th></tr>
<tr><td>3D-Printed Parts</td><td>Dog Head</td><td>Dog Bone</td></tr>
<tr><td>Laser-Cut Chassis</td><td>Chassis Base Plates</td><td>Chassis Bars</td></tr>
<tr><td rowspan="2">Circuitry</td><td>Pin Input for Sensors</td><td>Pin Output for Motors</td></tr>
<tr><td colspan="2">Building the physical circuitry and wiring (both)</td></tr>
<tr><td>Physical Assembly</td><td>Spray Painting and Hot Gluing the Chassis</td><td>Organizing the Arduino, Breadboard, and Battery Pack on the Chassis</td></tr>
<tr><td rowspan="4">Algorithm</td><td>Reading Sensor Inputs and Calculating Distance</td><td>Outputting Individual Motor Logic</td></tr>
<tr><td>Creating Robot Move States</td><td>Creating Motor States</td></tr>
<tr><td colspan="2">Deciding ranges for robot to move forward and backward (cm)</td></tr>
<tr><td colspan="2">Working on logic for turning left and right</td></tr>
</table>

Video of Working Demonstration

You can see a video demonstration of our treat-following robot in action in the ZIP archive attached with this file inside the folder named “Pictures and Videos”. There will be two videos, one with the original audio named “Working Demonstration.mov”, and the other with a voiceover explaining how the robot works named “Working Demonstration with Voiceover.mp4”.