Designing an Electric Car Toy on Our Own: A Human-Following Robot Using Pixy Cam Visual Detection

Chunhao Zou

September 28, 2018

Abstract:

In the paper, I will briefly discuss some of the existing approaches to enabling a robot to follow a targeted person. An alternative approach by employing a Pixy2 CMUCam5 Image sensor and Arduino will then be introduced. I will document the steps by which I made the robot, recount the problems I faced and illustrate the ways I tackled them. Finally, I will present a review of how to improve the robot in the future and discuss its potential applications.

Keywords:

Human-following, Robot, Pixy2 CMUCam5, Arduino, Microcontroller, Image Processing, Object Detection

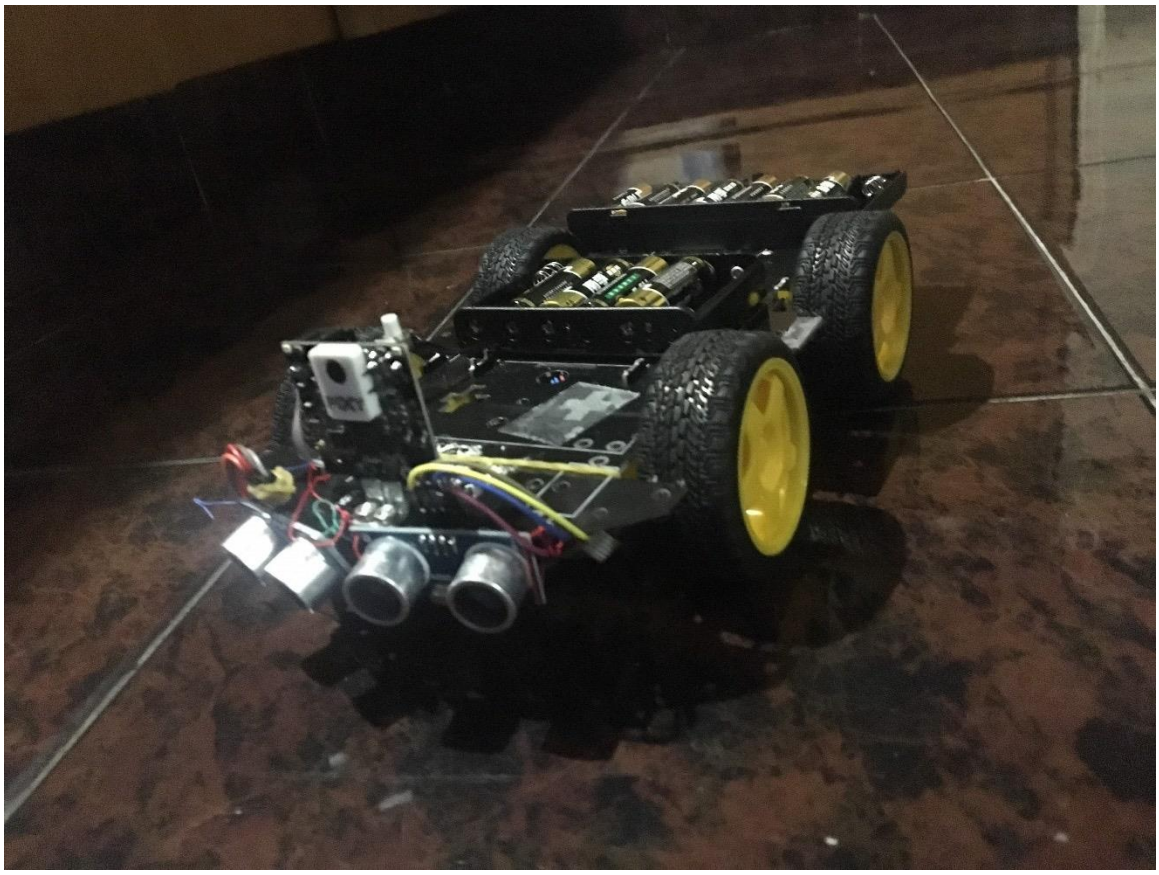

Figure 1: Prototype Robot

1. Introduction

Human-following robots have greatly attracted developers and researchers in recent years due to their potential application in work, manufacturing, and recreation. For instance, in 2016, a Californian tech company launched a suitcase that could automatically follow travelers [1]. This

year, Japanese multinational company TDK has created a robot plant pet with the ability to accompany a person and “offer emotional support” [2]. There are multiple methodologies to achieve the "following" functionality, such as utilizing ultrasonic transducers [3], infrared positioning [4], or motion detection [5]. Yet with the invention of the Pixy Camera, there exists a simpler and more graceful means to make a robot pursue a person, which will be elaborately documented in the paper.

The designing process of the robot involves teaching the camera to recognize an object, connecting the camera, the motors, the ultrasonic sensors and other components to Arduino, and then designing an effective algorithm to handle the data that the Pixy sends and to drive the motors in an appropriate way. Some functions like object avoidance (using ultrasonic sensors) should also be added so that the robot can be used in real environments.

I consider that a successful design should fulfill the following requirements:

1. Ability to track the target person in a large variety of environments without interference from other people/object/variables.
2. Ability to steer, proceed and move backward in order to follow the person
3. Ability to avoid some obstacles when approaching a person

To meet all of the requirements is not a simple task. During the process of making the robot, some issues and difficulties emerged, which I managed to address through multiple means and will be explained below.

2. Existing Positioning Methods

In this section, I will briefly summarize the mechanisms behind ultrasound-based, infrared-based and Kinect-based human following robots. Understanding how traditional methods work is essential. By evaluating the pros and cons of each method, we are able to develop a better model which can effectively address the problems they each have.

The main differences between these methods are ways to determine the relative direction of the person to be followed. Such a method of positioning is crucial because it tells the robot where to go.

2.1 Ultrasound-based Positioning

Ultrasound is sound waves with frequencies higher than 20KHz, which is beyond the threshold of human hearing. Because of the high frequency, the wavelength is extremely short, indicating its capability to travel through small spaces with little diffraction while keeping its direction unchanged. Therefore, ultrasound has become a popular way to measure distance.

Figure 2: HC-SR04 Ultrasonic Sensor
Image by www.botshop.co.za

A traditional ultrasonic sensor (Figure 2) contains an ultrasonic transmitter and a receiver. The transmitter emits ultrasound, which is reflected back when it

encounters an object, and the receiver detects the echo. The time it takes the sound to travel back and forth is measured. An ADC (Analog-to-digital Converter) converts analog data measured from the sensor to the microcontroller as digital data. Then based on the speed of sound in air, we can calculate the distance between an object in front of the sensor and the sensor itself.

According to [6], a simple way to employ ultrasonic sensors involves placing three ultrasound sensors at the front of the robot. Specifically, one is placed in the middle, and the other two on the left side and the right side. The sensors emit ultrasound in three directions, and then each sensor can measure the distance between itself and the object in its direction. The three distance values will be compared. In most cases, the direction of the sensor whose distance measurement is the shortest indicates the direction of the person. In this way, we can determine which direction the person is at and act accordingly.

Such kind of method is effective when the environment is relatively simple since the person is usually the closest "object" to the sensors. Yet it is still impractical in real life since an ultrasonic sensor can only detect things in front of it indiscriminately, which means it cannot tell what exactly the thing is, or whether it is the target person. Therefore, it is very likely that the robot will follow the wrong thing.

There is a way to improve ultrasonic detection, which utilizes a slightly different way to perceive the direction of the target person. Instead of working as one piece together, an ultrasonic transmitter and an ultrasonic receiver can be separated [3]. At least two receivers are installed on the robot. The transmitter, now called a "beacon", is carried by the person to send signals to the receiver. The receivers do not measure distances according to the time it takes ultrasound to leave and return to the sensor any more, but instead estimate the distance according to the magnitude of ultrasound signal detected. The closer the receive is from the beacon, the larger the signal. The direction of the receiver whose signal magnitude is the most intense is the exact direction of the person.

2.2 Infrared-based Positioning

Infrared light is electromagnetic radiation with a wavelength longer than that of visible light. It has a wide range of applications in our daily lives. Infrared LEDs are used to emit infrared light. When one LED is turned on and off quickly according to established patterns, useful information can be encoded and thus transmitted.

The general principle of infrared-based positioning is similar to that of ultrasound beacon positioning. Two infrared receivers are installed on the robot. The person holds the infrared beacon to send signals to the infrared receivers. To separate the receivers, a PCB sheet or any other equivalent will be placed between them so that they can only detect IR signals from one side, avoiding interference. If, for example, the receiver on the left detects the IR signal from the person, then the robot knows the person is to the left and will proceed leftward [4].

2.3 Motion Detection Based on Kinect

Kinect is a motion sensing device created by Microsoft, which provides a way to sense the position of human body parts. With the appropriate libraries installed and Kinect connected, a single-board-computer (e.g. Raspberry Pi), can receive the data from Kinect, which contains the

coordinate of the torso, the arms, the legs, etc. of the person [5]. With appropriate software, the robot can analyze the data and decide the direction it should go.

The Kinect method does not involve much work in microelectronics but instead is heavily software oriented. It provides a much more convenient way than the methods mentioned above, but cannot be employed in simple microcontrollers.

3. Design Approach Using Arduino and Pixy2 CMUCam5

According to Pixy2 Documentation, Pixy2 Camera (Figure 3) is a "small, fast, easy-to-use, low-cost, readily-available vision system" developed by the Carnegie Mellon Robotics Institute and Charmed Labs that can learn to detect objects that people teach it. Pixy has an integrated and independent processor which uses a "color-based filtering algorithm" to detect objects, making visual detection simple and convenient.

Figure 3: Pixy2
Image by docs.pixycam.com

I decided to use this method for multiple reasons. First, Pixy2 is easy to connect to Arduino since a cable for Arduino is already available. Secondly, Pixy2 is convenient because it only sends useful data, rather than dozens of megabytes of images, to Arduino, so coding is simple and easy. When it detects a target object matching a specific color pattern, it sends usable data containing the ID, the position, and the size of the object. The position data contains an X and Y coordinate which correspond to the position of the pixels where the object is located. Thirdly, Pixy2 can precisely track an object at 60 frames per second, which means no matter how fast the object is moving, Pixy2 will not lose track of it. Pixy2 can also assign index numbers to multiple objects with the same color, so Pixy2 will not confuse them and will instead assign "IDs" to them. Lastly, Pixy2 also has a color code detection technique. A color code is a juxtaposition of different colors (usually two colors) and can be regarded as a signal conveying specific meanings. With color codes, we can develop some useful functions such as commanding the robot to start, stop, turn on the LED or sound an alarm.

3.1 Configuration of Pixy2

Pixy can learn to recognize things quickly. We can either press the button and aim the camera at the object we want it to recognize, or connect it to the computer and select the target object on screen. In order to let it detect a person to be followed, I considered teaching it to recognize my T-shirt, which has a bright orange color and is easily distinguishable from the surrounding environment. Automatic white balance and automatic exposure were enabled for the camera to adapt to variable lighting conditions.

Sufficient light is a prerequisite for effective teaching and tracking. It is recommended that we teach objects in the day rather than at night, and outdoor rather than indoor. Natural light sources such as light from the sky are better than lights at home such as LEDs and incandescent lights because it is more intense and usually a larger proportion of the object surface is illuminated. When Pixy2 is taught in an ideal environment with sufficient light, it not only can still recognize

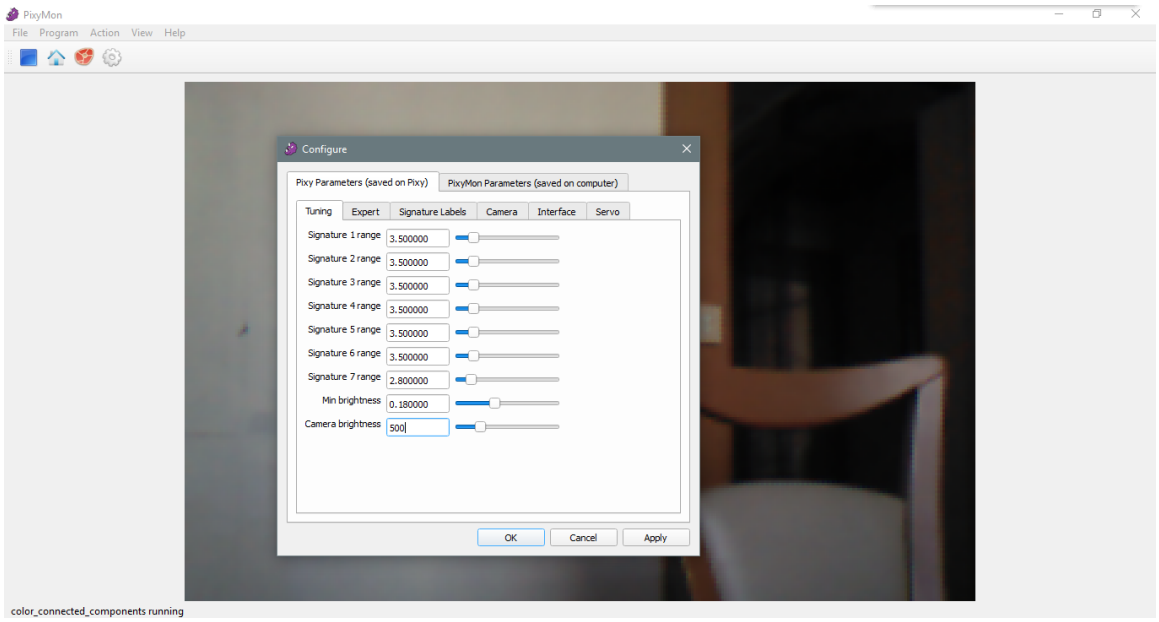

Figure 4: User Interface of PixyMon

the same object in an environment with weaker light (e.g. indoor environment) but also can better distinguish objects with similar colors. On the other hand, if Pixy2 is taught in a room with only a dim lamp on, it can easily confuse objects with similar colors.

Several parameters should be adjusted in order to improve the accuracy of recognition, and the work is done on PixyMon (Figure 4), a configuration utility that can run on Windows, MacOS, and Linux. Among all the parameters, signature range, basically the inclusiveness of color detection, is the most crucial. If we set the range too high, there can be some “false positives”, meaning something with similar colors will be considered as the target object. On the contrary, setting the range too low can cause Pixy2 not to detect the object we intend, or to “detect the object intermittently”. In an environment with intense light, objects with different colors can be perceived by the camera easily, so even if we set the range high (about 3.50), we can still perceive things accurately. Yet in darker environments, things with similar colors appear almost the same, so we have to adjust the range a little bit lower (to about 3.20) to avoid false positives.

Camera brightness should be set at a value lower than the default value (default 65, configured at 50) since overexposure can drastically reduce detection accuracy. Note that even though an environment with intense light is recommended, the camera brightness itself should be considerably low to avoid pale images.

When Pixy is connected to the computer and detects target objects, a block, or a rectangle surrounding the object in the image, will appear on screen. When there are multiple objects detected, there will be multiple blocks as well. Sometimes when the light condition is not ideal, or when the object is not evenly illuminated (e.g. when there are too many shadows or overexposures), one object could be perceived as multiple things, which means there could be two or more blocks appearing at the same time that surround different parts of the object. To avoid this mistake, we need to merge adjacent blocks to form one block, which is achieved by adjusting the value of “max merge distance”.

Finally, in order to ignore minor objects with the same color, Pixy2 should be set to detect only one block per signature. In this way, Pixy can always focus on my clothes and will not be "distracted" by some other objects with similar hues.

3.2 Other Kits for the Project

In the project, some other components play a vital rule too. The robot needs four geared DC motors which operate between 3V to 7V and have a maximum no-load speed of 290 RPM (at 7V). Four wheels, each 66 millimeters in diameter, are also employed. Therefore, the maximum speed of the robot will be approximately 1.0 m/s.

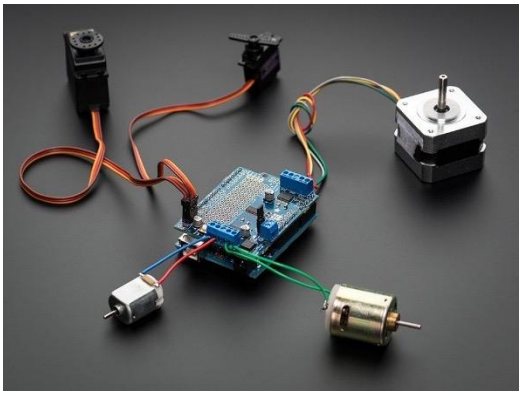

Figure 5: Adafruit Motor/Stepper/Servo Shield for Arduino v2 Kit - v2.3
Image by www.adafruit.com

To effectively control the motors, I utilize an Adafruit Motor Shield (Figure 5), which can handle at most four DC motors. The basic principle of the motor shield is the same as that of an H bridge, which can switch the polarity of a voltage applied to our motors, so as to make the motors run forward or backward (Figure 6).

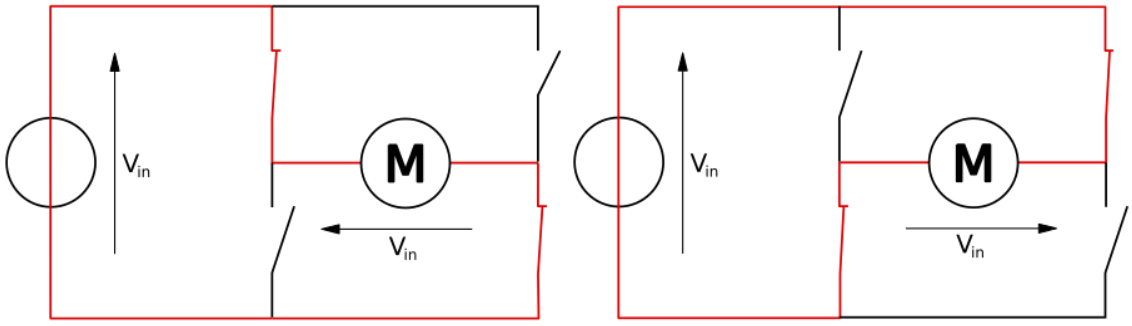

Figure 6: The Two Basic States of an H Bridge
Image By Cyril BUTTAY (own work, made using inkscape), via Wikimedia Commons

Two HC-SR04 ultrasonic sensors (Figure 2) are placed on the front left and front right side of the robot respectively and measure the distances between them and obstacles. The data is sent to Arduino and will help determine how to avoid the obstacles.

3.3 Algorithm

An effective algorithm was designed for the project. When we power the robot with a 9V battery, the robot will initialize the pixy camera, assign ports to the motors, and then start working. Arduino receives data from Pixy2 continuously, which tells it whether the target person is detected, and, if true, the x coordinate of the person. The robot will determine which "intended" action to take according to the value of the x coordinate. When there are no obstacles, if the value is less than 110, which means the person is to the left side of the robot, the robot will turn left by setting the right motors to spin faster. If the value is larger than 206, the robot will turn right.

When the value is between 110 and 206, which indicates that the person appears in the middle of the image, the four motors will all be set to run forward and the robot will proceed forward in a straight line. If the person is no longer present, the four motors will stop spinning. The actions like turning left, proceeding, turning right, and stopping corresponds to specific codes, which are integer constants and are represented by all-caps names. Seven available actions are shown in the table (Figure 7).

Code Name	Value	Description
RIGHT	0	All motors are on, but the left-side motors spin faster so that the robot can turn right
LEFT	1	All motors are on, but the right-side motors spin faster so that the robot can turn left
FORWD	2	All motors are on so that the robot can move forward
BACKWD	3	All motors are on so that the robot can move backward
STOP	4	All motors are off
BACKWD_RIGHT	5	Only left-side motors are turned on so that the robot can reverse to the right
BACKWD_LEFT	6	Only right-side motors are turned on so that the robot can reverse to the left

Figure 7: Available Actions

I use the word “intended action” because the true action is not so simple as what is described above. There can be various obstacles in the way, so robots need to make wise decisions to avoid them. A void function, getDist(), is employed to read data, named rightD and leftD, from the ultrasonic sensors. The values are measurements of the distances (in cm) between obstacles and each sensor. A function, getTrueAction(int intended), does the decision making by evaluating both the intended action and the condition of the surroundings. For instance, when the obstacle is too close (<10 cm) to either sensor, the robot will reverse either to the left or to the right based on which sensor is the closest to the obstacle. When either or both sensors are close to the obstacles (<20 cm), but not too close (<10 cm), the robot is likely to stop, turn left or turn right, according to various conditions. For instance, when the intended action is to move forward, the value of leftD is between 10 and 20, and the value of rightD is larger than 20, the robot will not move in a straight line but instead avoid the obstacle by turning right. Finally, a function, moveRobot(int actionCode), adjusts the direction and the speed of the motors according to specific codes returned by getTrueAction.

Flowcharts showing the overall decision tree of the robot (Figure 8) and the function getTrueAction (Figure 9) are presented.

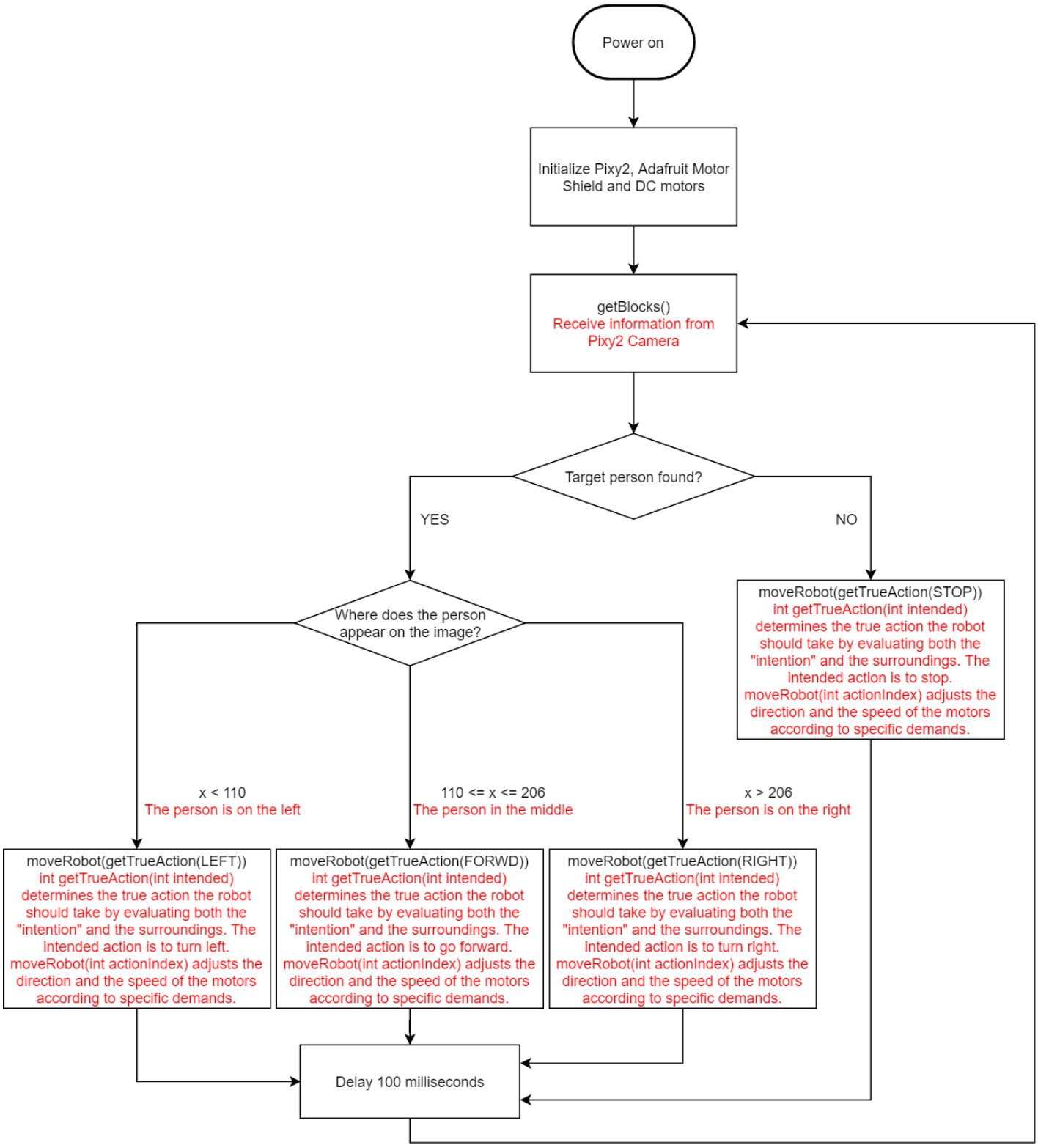

Figure 8: The Overall Decision Tree of the Robot

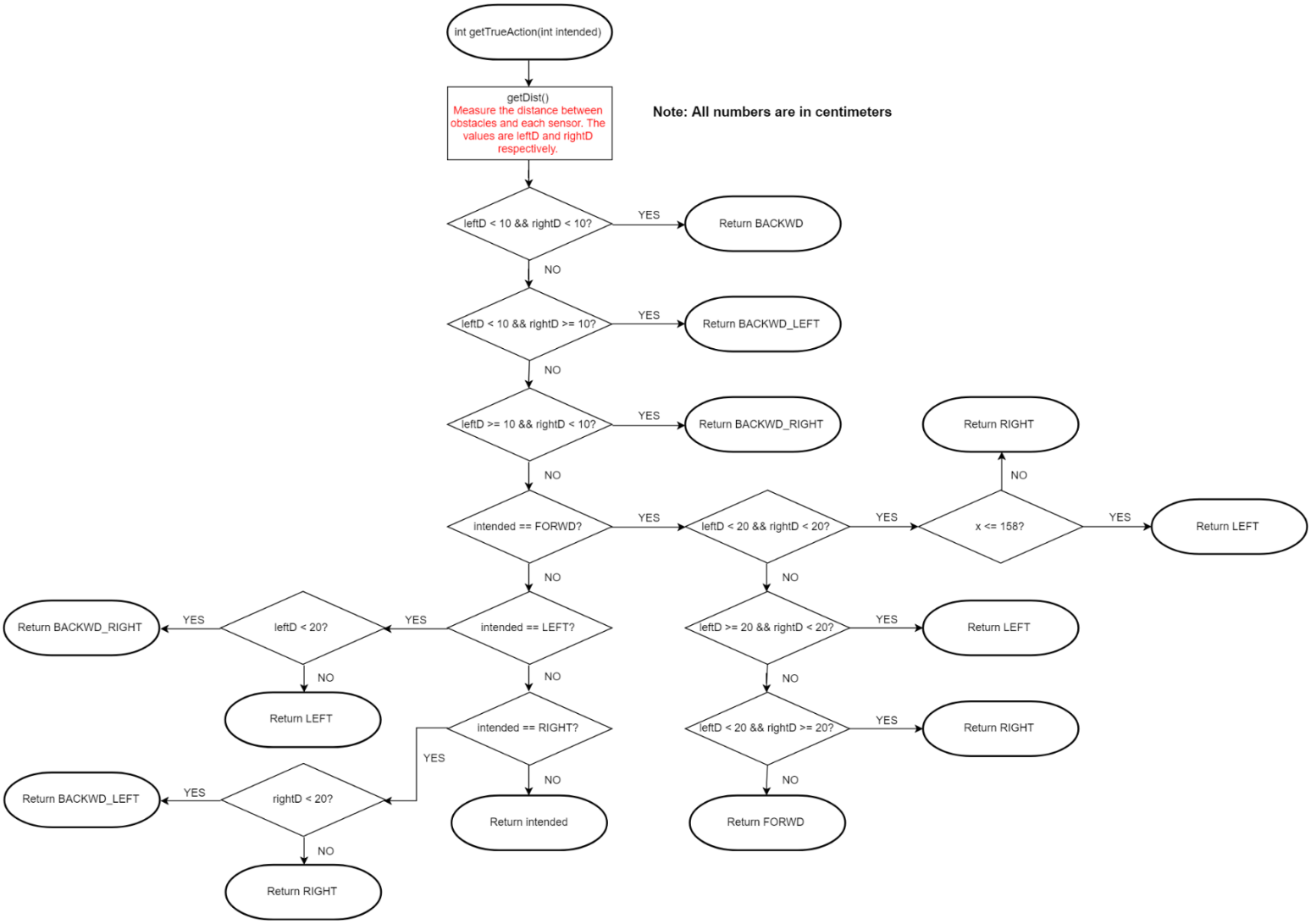

Figure 9: Function getTrueAction

4. Tests, Problems and Solutions

To experiment with the algorithm I developed, I devised three tests: an indoor test, an outdoor test, and an object avoidance test. The indoor test was run on an open space in my living room. The outdoor test was run on an outdoor space in my neighborhood on both sunny days and cloudy days. The object avoidance test was run in both places. The testing videos are available at https://www.robotchz.tk.

The indoor test evaluated the robot's ability to keep track of a person in a dim environment, and during the test, the robot performed well in keeping pace with the person and could follow a person walking from one room to another with different lighting conditions. The outdoor test worked well on cloudy days, and the robot capably followed one person without losing track or being distracted by the surrounding environment. On sunny days, the robot would sometimes lose track of the target person if the camera directly faced the sun (during sunset, for example),

Figure 10: Outdoor Test on a Sunny Day

but was generally still good at staying focused on the person. In object avoidance tests, however, the robot could only avoid small obstacles. If there was a large obstacle in front of it, the robot would repetitively move forward and backward in front of the obstacle and would be unable to proceed. The condition indicates imperfection with the algorithm I employed. The left and right side of the robot could still collide with obstacles due to insufficient measuring angles of the ultrasonic sensors (in this case, the measuring angle of HC-SR04 is only 15 degrees), so sensors that can “look around” (with the help of servo motors) or can measure things in a wider angle are needed. Also, ultrasonic sensors cannot detect flat objects not facing them directly, since echoes emitted by the transmitters cannot bounce back to the receivers under this circumstance. Therefore, sometimes the robot is still incapable of sensing and avoiding obstacles positioned in front of them at certain angles.

During the process of robot building and testing, some other problems, either large or small, occurred, hindering my progress. However, through continued endeavor, I managed to tackle them and improve the overall performance of the robot.

The connection of jumper wires to the Adafruit Motor Shield was problematic because the size of the pins on the motor shield was too small, resulting in intermittent malfunctions of the ultrasonic sensors. When the ultrasonic sensors lost connection, the data read from the analog input pins

would fall to zero. The robot would then mistakenly think it was facing an obstacle “0 centimeters from its sensors”, and move backward. I tried different approaches to solve the problem and finally discovered that inserting copper wires into the connectors provided a simple way to improve the connectivity.

The most crucial problem I encountered was power management.

When the robot was powered by the 9V battery, unexpected things could happen:

1. If the speed of the motors was set low (<=110), the robot would work properly.
2. If the speed of the motors was set high (=170), the robot became “sluggish”. Not that the speed of the motors was slow, but that the robot was unable to process the information sent by Pixy2 and the ultrasonic sensors quickly. For example, when a target object was placed in front of the robot and then removed quickly, the robot would not start and stop quickly. instead, it would move forward for an extended amount of time, sometimes even tens of seconds.

This problem occurred due to insufficient power provided by the batteries since the battery was running low. More specifically, with insufficient power, the microcontroller was unable to operate and do calculations consistently. Changing batteries worked, but the robot would only operate normally for approximately one hour before this power problem returned. This is because the robot draws too many amps, draining the battery quickly. To fundamentally address the problem, I separated the power for Arduino and the four motors and used a 12V battery case for the motors.

5. Potential Applications

Human following robots have a large variety of potential applications. Aside from the automatic following suitcase mentioned at the beginning of the paper, such technology can also be employed in a shopping mall. A shopping cart with human following feature enhances efficiency in transportation and renders the shopping experience less grueling and more convenient [7]. In military occasions, the robot can carry ammunitions and medical supplies and reduce the burden on the soldiers, serving as a digital “mule” [8]. A Battlefield Extraction-Assist Robot (BEAR) developed by Vecna can even “lift injured soldiers and get them to a safe location for medical assistance” [9]. A drone with human following functionality can play the role of “eyes in the sky”, serving as a camera and shooting marvelous 4K videos [10].

The human following robot I designed has some specific potential uses in recreation. For instance, it could be a good toy for young children, who are curious about everything that can move and are very likely to interact with it, either by touching it, running after it, or running away from it (which will cause the robot to catch up). The robot may stimulate the interests of the kids, help them have fun, and, through the process of playing, help maintain a healthy mindset. Employing the same techniques, but to different targets, human following robots could become “pet following robots” and potentially be good toys for pets as well. Pets that do not get enough stimulus and exercise could keep fit physically and mentally when engaged in robot play.

6. Conclusion

In this paper, I designed a human following robot using a novel tracking technique and tested how it works in the real world. Although the design is generally simple, it proved effective. It performed well in various lighting conditions and placement conditions, and the robot generally demonstrated great ability to keep track of the target person. It also exhibited the capability of avoiding obstacles, though it can be further improved with more work.

There can be improvements made to the robot in the future to make it smarter and more powerful. Aside from improvements in the obstacle detection algorithm and better apparatuses for measuring distances, which are mentioned above, color codes are still not employed in this version of my robot, so they can be implemented for more precise object recognition. A color code is a signal conveying specific information, which we can use to develop useful functions such as commanding the robot to start, stop, turn on the LED or sound an alarm. Utilizing this technique could greatly improve the overall user experience of the robot since people could more easily engage in active communication with the robot.

This prototype robot is successful in realizing the target functionality we defined at the start of the research project and points the way to further improvements.

References

[1] M. McFarland, "This suitcase will follow you home like a puppy," CNNMoney, 18 Oct 2016. [Online]. Available: https://money.cnn.com/2016/10/18/technology/suitcase-autonomous-follow/index.html. [Accessed 7 Sep 2018].

[2] K. Marchese, "bonsAI is the smart robotic pet, agony aunt and trendy plant all in one," Designboom, 6 Apr 2018. [Online]. Available: https://www.designboom.com/technology/bonsai-smart-plant-ai-tdk-03-06-2018/. [Accessed 8 Sep 2018].

[3] R. Motisan, "Building a robot – Make the robot follow you," PocketMagic, 7 Feb 2013. [Online]. Available: https://www.pocketmagic.net/make-the-robot-follow-you/. [Accessed 27 Aug 2018].

[4] S. Afghani and M. I. Javed, "Follow Me Robot Using Infrared Beacons," *Academic Research International,* vol. 4, no. 3, pp. 63-72, 2013.

[5] S. Dzwończyk and M. Twardak, "Human-Following Robot with Kinect," Hackster.io, 10 Aug 2017. [Online]. Available: https://www.hackster.io/turtle-rover/human-following-robot-with-kinect-efb3cd. [Accessed 10 Sep 2018].

[6] A. Chalke, A. Bhor, P. Bhoite and U. Jadhav, "Follower Robotic Cart Using Ultrasonic Sensor," *International Journal of Engineering Science and Computing,* vol. 7, no. 3, pp. 5794-5797, 2017.

[7] Y. L. Ng, C. S. Lim, K. A. Danapalasingam, M. L. P. Tan and C. W. Tan, "Automatic Human Guided Shopping Trolley with Smart Shopping System," *Jurnal Teknologi,* vol. 73, no. 3, pp. 49-56, 2015.

[8] J. Pappalardo, "The Day the Marines Met Their Robotic Mule," Popular Mechanics, 12 Nov 2013. [Online]. Available: https://www.popularmechanics.com/military/g1345/the-day-the-marines-met-their-robotic-mule/. [Accessed 28 Sep 2018].

[9] S. McCormack, "Military Robots Ease a Soldier's Load," KVH Mobile World, 8 Apr 2013. [Online]. Available: https://www.kvhmobileworld.kvh.com/military-robots-ease-a-soldiers-load-2/. [Accessed 28 Sep 2018].

[10] N. Statt, "Skydio's AI-powered autonomous R1 drone follows you around in 4K," The Verge, 13 Feb 2018. [Online]. Available: https://www.theverge.com/2018/2/13/17006010/skydio-r1-autonomous-drone-4k-video-recording-ai-computer-vision-mapping. [Accessed 27 Sep 2018].