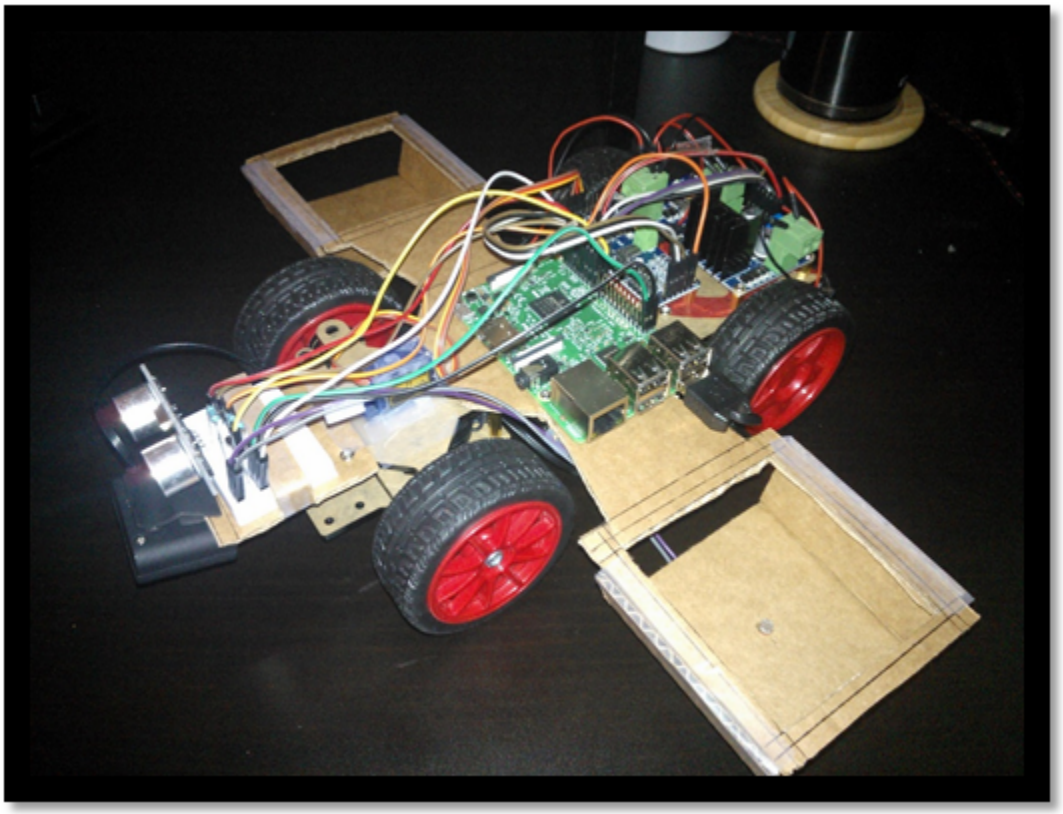
Bitcoin Tim - Delivering Automation

Jiaming Hu
jih189@ucsd.edu

Song Young Moo
yus086@ucsd.edu

Jose Guan Zhou
gzjosef@gmail.com



CONTENTS

I. Abstract

Audrey's is conveniently available for students studying on the second floor of Geisel Library. However, it is situated at a significant distance away from the popular study group tables, where most students are located at. Bitcoin Tim is a proposed solution to this predicament by delivering drinks and meals to customers who submitted an order to Audrey's. Tim will accomplish its mission through the use of the OpenCV library and multiple sensor modules in order to recognize the path it needs to follow to reach each customer, the presence of drinks, meals, and obstacles, respectively.

II. Introduction

In recent years, UC San Diego has received a increase of freshman and transfer applications[1]. This has resulted in a growing population in its student body[2] and density of students present in Geisel Library, a popular meeting and studying location for students. Audrey's is a coffee shop opened on the second floor of the structure in order to accommodate students by providing them a nearby store to purchase drinks and light meals. However, Audrey's is located at one corner of the floor while the tables and computers that are regularly used by students to work, study, and collaborate on are located at the opposite corner of the same floor. This is inconvenient for those of whom do not wish to leave their items unattended, walk away from their work, or wait in line in case of there being many customers ordering at the shop. In order to address these issues our team has developed a prototype solution, Bitcoin Tim, an embedded system device tasked with obtaining drinks and meals prepared by the baristas of Audrey's, deliver to customers their ordered products, and returning back to the coffee shop, avoiding any obstacles present on its track. For the purpose of this primitive implementation, Tim will follow a path laid out on the floor, recognizing each table stop through a red marking on the path, and stopping on the occasion of an obstacle present on the path and resuming its travel once the obstacle is no longer present. Students and customers will be able to place an order to Audrey's through a proprietary application, specifying their choice of drink or meal and their current table number. A barista will then prepare said order and place it on Tim, who, once it has recognized that all ordered items have been prepared and placed, will begin to deliver. Once Tim has reached a customer's table, it will wait until said customers has taken their product, where it will then continue to deliver the remaining orders. Once all items have been delivered Tim will return back to Audrey's, ready to repeat the process for new incoming customers. This proposed solution would facilitate students by allowing them to order at the comfort of their seats without having to leave their current location.

III. Technical Materials

- Hardware Specifications
  - Raspberry Pi 3 Model B - central module, controlling both DC motors for the wheels located at the back, servos (PWM), and submodules which include the USB camera and ultra sonic sensor. Communicated wirelessly with the use of the program VNC server to a remote computer running VNC viewer, which would execute the driver program written in Python.
  - Step Down DC-DC Converter Module - converted output voltage to an appropriate amount for each module in the system, e.g. Raspberry Pi, DC motor module, and servo controller.
  - DC Motor Driver - module whose primary purpose is to drive the two DC motors attached to the back wheels to move Tim forward and backwards.
  - Servo Controller - module that sends PWM (Pulse Width Modulation) signals to the servo motors in order to control the angle of rotation for the front wheels and regulating the speed of the back DC motors. This module was utilized in order to have better control of the speed in which Tim would travel at.
  - Camera Module - USB connected camera which detects the black line and red marks that simulate the path to follow and the table stops, respectively.
  - Ultra Sonic Module - module that will allow Tim to recognize whether an obstacle is present on the path, which will cause Tim to stop moving and will resume its deliveries once the obstacle is no longer present.
  - Batteries - Two 18650 batteries used to power all modules, motors, and the Raspberry Pi, emitting 3.7V.
  - Plastic Head and Platform - 3D printed parts attached to Tim in order to hold the front USB camera and the cup holders. The models were generated through Google SketchUp.
  - 2 DC Motors - motors to control the back 2 wheels that go either forward or backward
  - Servo Motor - motor that controls the direction of the front wheels which determine the overall traveling direction of the entire system
  - Photo Resistor - small photo resistor connected to a breadboard which is then connected to the Raspberry Pi, placed at the bottom of each cup holder in order to determine whether an object was placed in the holder. If an object is present, the photo resistor will detect low light input, otherwise it will detect high light input if no object is present.

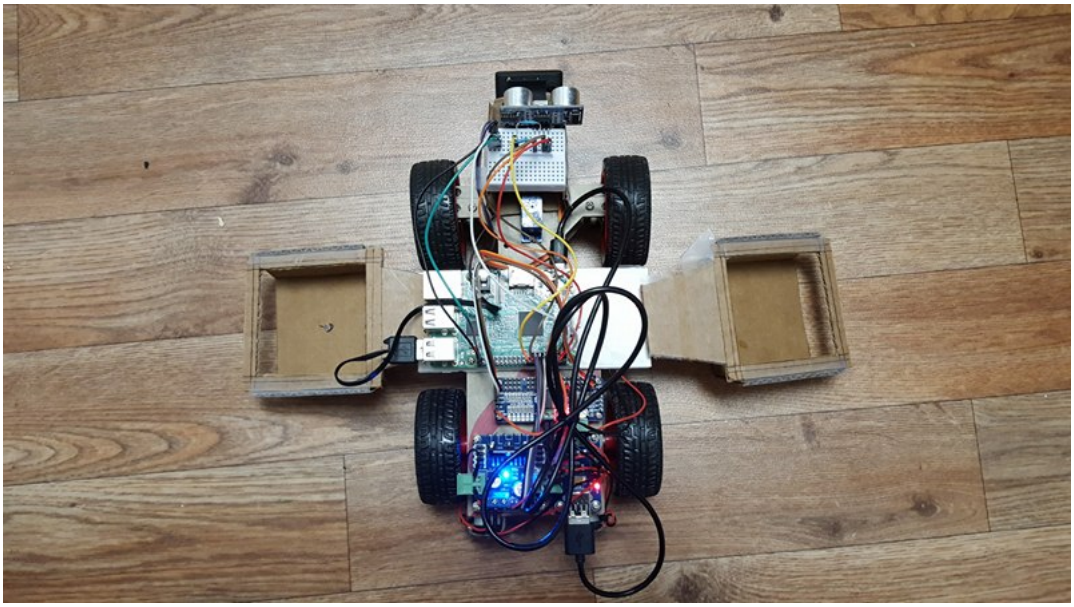


Fig. 1. Top view of new implementation of Bitcoin Tim

- Software Specifications
  - OpenCV - this library provided the capabilities for Tim to move, recognize the laid out path and red-marked spots, and detect whenever there was an obstacle present ahead. OpenCV and most programs that drive Tim's functionalities are implemented in Python 2.7, as the Raspberry Pi utilized for Tim also ran in the same language.
  - Android Studio - a simplified Android application was created as a sample interface that future customers can use to submit orders. At its current state, the application requires the user to input the IP address of Tim in the network it communicates through, though this can be omitted if a stable and consistent network is established for Tim to connect through. After the address has been submitted, the user can inform Tim at what table they are currently located at by inputing their assigned table number. Once Tim begins delivering all orders, it will calculate how many stops it needs to pass in order to reach the customer's table. At the moment, no process of determining which order belongs to which customer has been implemented, nor a menu of available items to order from has been created.

IV. Milestones

1) Obtaining the necessary components and assembling Tim - in order to know what tools to utilize and the capabilities if the embedded system, research must be done on both the board and components, and assembly must take place

   Revisions and Issues - in the beginning our team decided to use an Arduino to be the main module to control the system due to its low price and simpler environment setup. However, our team had more experience working with the Raspberry Pi and were aware of the capabilities of the library OpenCV for this module. Furthermore, we sought for a simplistic car chassis as we only needed the system to move forwards, backwards, and sideways. Such model was found on Amazon, which was being sold by the company Mood, and used a Raspberry Pi as its central model. The initial model did not allow the batteries to be connected to more than one module, thus we purchased a Step-Down DC-DC Converter module. Lastly, in order to control the speed of then back motors, a DC Motor Driver was purchased, as Tim would need to travel at a speed in which the items it would carry would not fall over.

2) Testing functionality of components - each component will be checked to see that they are working appropriately. Simple programs will be made in order to determine if all servos and motors are working

   Revisions and Issues - most of the components were tested early on during the development of the system, although the photo resistors were left towards the end due to the necessity of printing the chassis first in order to fit the resistor later on. The USB camera component caused the highest amount of difficulty of use, as the frames per second recognized by the system was low and our initial tests required the system to recognized certain aspects of colors and shapes with as low latency as possible. Initially it was proposed that each mark that simulated a table stop should be in a circular

shape colored in red. However, as the camera was not placed on a flat surface, was close to the ground, and shadows would interfere with the camera input, the marks had to be revised. Instead, it was agreed upon that the markings would instead be of any shape but had to be colored in red, an easily distinguishable color, where the system would filter out all colors it read from the camera but red, and once such color was present it would recognize it as a table stop. The photo resistor and ultra sonic sensors required a breadboard in order to be implemented, which cause slight difficulties in space as initially our team did not account for a breadboard to be present on the system. This was addressed when constructing the 3D model for the chassis.

3) Research available tools for implementation - research and determine which tools are necessary for the project, such as libraries that will be loaded into the system. Furthermore, IDEs will be tested and practiced on in order to become familiar with the environment

   Revisions and issues - since our initial assumption was to utilize an Arduino, we began experimenting with the Arduino IDE. However, as soon as we decided to switch to the Raspberry Pi, we researched on libraries and tools available for use in the Pi system. In order to manage the vision component of the system, our team researched and began experimenting with OpenCV, a library that allowed us to read the input from the USB camera and ultra sonic sensor. As most of our team members had experience developing on Python, testing the components with this library was not difficult. It was at this phase that our team recognized the limitations of the reaction time of the system, as the camera could only record around 10 to 15 frames per second, which introduced a latency in reaction, especially when Tim was moving.

4) Programming basic movements - the first step into the project will be to program movement into Tim. This milestone aims to let Tim know where to stop and when to stop

   Revisions and Issues - there were nearly no issues encountered in this milestone. Most of the concerns for this milestone was determining the appropriate speed in which the system should travel, where if it was too fast it would lose track of the line path, and if it was too slow it would not move due to the weight of the system. We calculated the necessary speed after doing some environmental tests.

5) Acquiring live data from input modules - receive and interpret data read from the photosensitive sensors and being able to determine if a cup is present

   Revisions and Issues - as noted above, the photo resistors were not tested until the end of the project, though they would not introduce new issues and were implemented with little to no difficulties. The USB camera, on the other hand, did introduce the presence of high latency. Its significance will be discussed in the following milestone. The ultra sonic sensor did not introduce any issues, though in order to implement it we required a breadboard, and had to determine the best position to place this breadboard that would not conflict with the other modules. Initially we wished to place it at the front of the system, but this would not provide any remaining space for the USB camera which was clipped to the front. Placing it at the bottom would also be inefficient as the ultra sonic sensor would always detect the presence of the camera and therefore the entire system would not move. In order to address this concern, we decided to create a 3D model that would place the breadboard at the top without having the ultra sonic sensor too far away from the front of the system, and the USB camera right below it with enough clearance from the ground.

6) Generate and conduct isolated capability tests - tests will be significant functionality segments of the overall duties of Tim. This includes following a path, stopping if an obstacle is present and moving once said obstacle is removed, having a cup above it must move after a certain amount of time after cup is removed

   Revisions and Issues - most of the issues encountered for this project were found after conducting the live tests. One major issue was developing the 3D model, which was not an initial milestone and instead a promise and desireable. The difficulty came from finding an available 3D printer that could generate the model in a short amount of time and was available. Some options considered were the 3D printer available in Geisel Library, the printing room in the MAE building, and a specialized company printer from General Atomics. The printing room in the MAE building was not available to students who had not taken a safety quiz that was offered to students in the Mechanical Engineering major, and since none of our team members were in that major we decided to find another option. The specialized printer was available as one of our team members had a friend who was currently working at the company and offered to send any model to be printed. However, any models that were sent to be printed would not be available within the same or the following day, as the printer often had many jobs being submitted by workers in the building. Instead, each job request

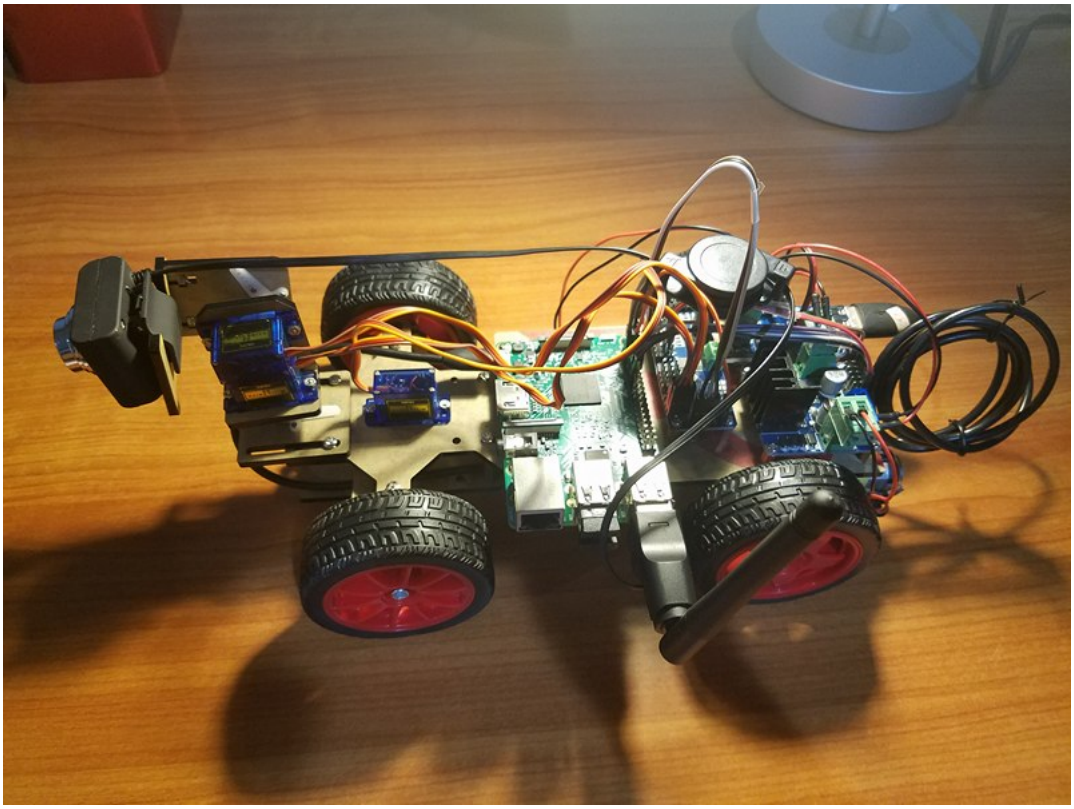


Fig. 2. Original version of Bitcoin Tim with no 3D printed chassis

would take approximately 3 to 4 days to be completed, and due to the nature of 3D printing, each model would shrink making all measured hole placements to be incorrect. This option was left out and instead replaced with the printers available in the library. The only issue encountered when using these printers was the fact that reservations were required and by the time our team decided to utilize them at the end of week 8, they were mostly all booked until the end of the quarter. Furthermore, each printer had a time limit of 3 hours, which made it impossible to print large items, such as the cup holders that would be added into the chassis. Thus, we prioritized the segments of the system that needed to be generated and chose both the head of the system that would hold the breadboard for the ultra sonic sensor and the USB camera, and the plate that would attach to the cup holders on the side of the system. The cup holders were made from cardboard cutout instead due to their large size.

After generating the 3D chassis, our next issue was in regards to the live input of the system from the USB camera and photo resistors. An issue we encountered when implementing the photo resistors was the light necessary for them to recognize that an item was not present in the cup holders. Our approach for recognizing whether an object was in the cup holders was relying on the nature that if an object was present, then the bottom of the cup holder would be devoid of light. On the other hand, once the object was not present the bottom would be bright once again. Therefore, by placing a photo resistor at the bottom of the cup holder would allow the system to recognized whether an item had been placed, which would signal Tim to not move if no item had been placed by a barista, or to stay until a customer had taken their product once Tim would reach a table stop. In order to resolve this we conducted all of our tests in brightly lit areas placed at the ceiling of the room in order for the resistors to recognize that there was no object present.

Early on our team realized the slow amount of frames captured by the USB camera. This was a major concern as the system would utilize these frames in order to determine which direction it would need to travel. The path laid out would be a black line which Tim would try to center when looking through the camera. If the line was too far to the right of the frame Tim would move to the right, and likewise if the line was too far to the left it would move the system to the left. If no line was recognized the system would move backwards until a line could be found and continue to follow it. With the slow amount of frames captured, the system would often fail to follow the line quick enough and would require to move backwards to recalculate the center of the path. Furthermore, if the speed of the system was set too high, Tim would easily lose track of the path and would have difficulties trying to make sharp turns on the path. Therefore all environmental tests required the paths to have a curving angle in order to prevent Tim from losing track

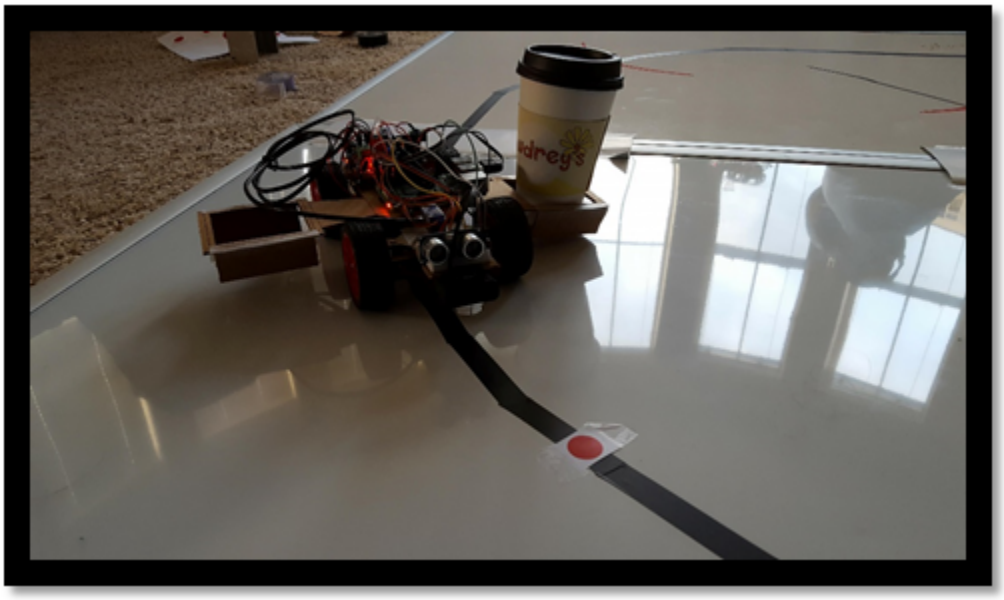


Fig. 3. Bitcoin Tim lays on top of a sample laid out path with a table stop marker placed further ahead

of it.

The other issue we found was caused by the positioning of the USB camera. Although there was enough clearance between the camera and the ground, the camera had a limited field of view. This made it difficult for the system to recognize a circle that was meant to simulate a table stop correctly. Instead, our team decided to conclude that once any red marking was present in the frame and would later leave counted as a table stop that had just passed by. This made it simpler for the system to get a consistent count of tables it had passed and made the overall process faster, as attempting to determine the presence of a circle in the frame caused exhaustive calculations.

7) Create a android application and send the order to Tim and run all simple tests again

Revisions and Issues - our team attempted to create a basic prototype Android application that demonstrated the ease of use for customers to order their drinks and meals. As the main priority for this application was a proof of concept, no menus or item selection were implemented into the application. Instead, the application requires the local IP address of the system. Afterwards, the user may enter the number of the table in which they are currently sitting at. The largest issue for this milestone was establishing the connection between the application and the system, as our initial tests on the school's network were impossible to conduct due to the firewall that denied our application access to the system. Therefore, all tests that utilized the Android application were conducted at a team member's house and the firewall of the system was disabled.

V. Conclusion

While many more improvements are possible on this system, the purpose of this project was to demonstrate a possible solution to a problem common to students partaking in group studies or who are inconveniently located away from Audrey's coffee shop. Overall, Tim was able to follow a predetermined path, keeping track on the presence of the items it had to deliver and any possible obstacles that may appear on its path. For future work, our team wishes to develop a more stable item handling chassis. Increasing the amount of items Tim has to keep track of would simply require more photo resistors to be connected to the system. Meanwhile, a camera with a wider field of view would allow Tim to generate a smoother travel path and lower the chances of path deviation. Further development on the application is also possible, adding support for a menu and item selection, and establishing a static IP address for the application to directly connect and communicate with Tim instead of having the user manually inputing the address. Lastly, for a more complex future feature, our team is interested in utilizing computer vision and have Tim learn and travel through the shortest path from any two points on the floor of the library, removing the need for a laid out path.

REFERENCES

[1] Christine Clark

http://ucsdnews.ucsd.edu/pressrelease/a_record_116452_freshman_and_transfer_students_apply_to_uc_san_diego_for_fa

[2] Gary Robbins

http://www.sandiegouniontribune.com/news/science/sdut-UCSD-enrollment-growth-2016may22-story.html