AutoVehicle

Khemar Bryan, Jan Yalda, Bilal Al-fanous (JBK.17)

March 31, 2017

AutoVehicle by JBK.17

Project website:https://github.com/Khemar1/khemar1.github.io

Declaration of Joint Authorship

We, Jan Yalda, Bilal Al-fanous & Khemar Bryan confirm that this work submitted for assessment is our own and is expressed in our own words. Our best effort was made to split this project equally. Jan Yalda worked on the hardware and database. Bilal Al-fanous worked on and maintained the web site. Khemar Bryan worked on the mobile application. We all collaborated to ensure that all parts of this project were able to connect and communicate with each other. All direct or indirect sources used are acknowledged as references.

Date: March 31, 2017

Approved Proposal

Proposal for the development of AutoVehicle

Prepared by Khemar Bryan, Jan Yalda, Bilal Al-Fanous
Computer Engineering Technology Students

Project Website : https://khemar1.github.io

Proposal Table of Contents

Executive Summary

As students in the Computer Engineering Technology program, we will be integrating the knowledge and skills we have learned from our program into this Internet of Things themed capstone project. This proposal requests the approval to build the hardware portion that will connect to a database as well as to a mobile device application. The internet connected hardware will include distance sensors and actuators for controlling the movement of the vehicle. The database will store the mapped area. The mobile device functionality will include remote controlling the vehicle and displaying the mapped area and will be further detailed in the mobile application proposal. We will continue to work together this Winter semester as we all built similar hardware last term and worked on the mobile application. The hardware was completed in CENG 317 Hardware Production Techniques independently and the application was completed in CENG 319 Software Project. These will be integrated together this semester in CENG 355 Computer Systems Project as a member of this 3-student group.

Background

The problem solved by this project is how to have a robotic vehicle that will be controlled as an RC car with a mobile application, and which can work independently as an autonomous car.

The Hardware which consists of the vehicle's chassis where all the parts are either connected or mounted on which includes the actuators, the H-bridge driver, the ultrasonic sensors and the brain of it all the Raspberry Pi 3. The vehicle's sensors are used to avoid obstacles that are detected and at the same time send data to be used to map an area that will be displayed in the mobile application. Through Wi-Fi and Bluetooth the ability to remote control the vehicle is possible using the mobile application.

We have searched for prior art via Humber's IEEE subscription selecting "My Subscribed Content" [1] and have found and read the following three articles which provides insight into similar efforts.

- Algorithm Fusion for Feature Extraction and Map Construction from SONAR Data(Ismail & Balachandran, 2015)
- SLAM for robot navigation(Temeltas & Kayak, 2008)
- An Open-Source Scaled Automobile Platform for Fault-Tolerant Electronic Stability Control(Katzourakis, Papaefstathiou, & Lagoudakis, 2010)

In the Computer Engineering Technology program, we have learned about the following topics from the respective relevant courses:

- Java Docs from CENG 212 Programming Techniques In Java
- Construction of circuits from CENG 215 Digital And Interfacing Systems,
- Rapid application development and Gantt charts from CENG 216 Intro to Software Engineering,

- Micro computing from CENG 252 Embedded Systems,
- SQL from CENG 254 Database With Java,
- Web access of databases from CENG 256 Internet Scripting; and,
- Wireless protocols such as 802.11 from TECH152 Telecom Networks.

This knowledge and skill set will enable me to build the subsystems and integrate them together as my capstone project.

Methodology

This proposal is assigned in the first week of class and is due at the beginning of class in the second week of the fall semester. My coursework will focus on the first two of the 3 phases of this project:
Phase 1 Hardware build.
Phase 2 System integration.
Phase 3 Demonstration to future employers.

Phase 1 Hardware build

The hardware build will be completed in the fall term. It will fit within the CENG Project maximum dimensions of 12 13/16" x 6" x 2 7/8" (32.5cm x 15.25cm x 7.25cm) which represents the space below the tray in the parts kit. The highest AC voltage that will be used is 16Vrms from a wall adaptor from which +/- 15V or as high as 45 VDC can be obtained. Maximum power consumption will be 20 Watts.

Phase 2 System integration

The system integration will be completed in the fall term.

Phase 3 Demonstration to future employers

This project will showcase the knowledge and skills that I have learned to potential employers.

The tables below provide rough effort and non-labour estimates respectively for each phase. A Gantt chart will be added by week 3 to provide more project schedule details and a more complete budget will be added by week 4. It is important to start tasks as soon as possible to be able to meet deadlines.

Labour Estimates	**Hrs**	**Notes**
Phase 1		
Writing proposal.	9	Tech identification quiz.
Creating project schedule. Initial project team meeting.	9	Proposal due.
Creating budget. Status Meeting.	9	Project Schedule due.
Acquiring components and writing progress report.	9	Budget due.

Mechanical assembly and writing progress report. Status Meeting.	9	Progress Report due (components acquired milestone).
PCB fabrication.	9	Progress Report due (Mechanical Assembly milestone).
Interface wiring, Placard design, Status Meeting.	9	PCB Due (power up milestone).
Preparing for demonstration.	9	Placard due.
Writing progress report and demonstrating project.	9	Progress Report due (Demonstrations at Open House Saturday, November 12th, 2016 from 10 a.m. - 2 p.m.).
Editing build video.	9	Peer grading of demonstrations due.
Incorporation of feedback from demonstration and writing progress report. Status Meeting.	9	30 second build video due.
Practice presentations	9	Progress Report due.
1st round of Presentations, Collaborators present.	9	Presentation PowerPoint file due.
2nd round of Presentations	9	Build instructions up due.
Project videos, Status Meeting.	9	30 second script due.
Phase 1 Total	**135**	
Phase 2		
Meet with collaborators	9	Status Meeting
Initial integration.	9	Progress Report
Meet with collaborators	9	Status Meeting
Testing.	9	Progress Report
Meet with collaborators	9	Status Meeting
Meet with collaborators	9	Status Meeting
Incorporation of feedback.	9	Progress Report
Meet with collaborators	9	Status Meeting
Testing.	9	Progress Report
Meet with collaborators	9	Status Meeting
Prepare for demonstration.	9	Progress Report
Complete presentation.	9	Demonstration at Open House Saturday, April 8th, 2017 10 a.m. to 2 p.m.
Complete final report. 1st round of Presentations.	9	Presentation PowerPoint file due.
Write video script. 2nd round of Presentations, delivery of project.	9	Final written report including final budget and record of expenditures, covering both this semester and the previous semester.

Project videos.	9	Video script due
Phase 2 Total	**135**	
Phase 3		
Interviews	TBD	
Phase 3 Total	**TBD**	
Material Estimates	**Cost**	**Notes**
Phase 1		
Raspberry Pi 3 Kit	$99.00	CanaKit
4WD Robot Platform	$42.94	Creatron Inc
L298N H-Bridge	$15.82	Creatron Inc
Mini Bread Board	$3.67	Creatron Inc
Jumper Wires(3-sets)	$6.86	Creatron Inc
HC-SR04 Ultrasonic Sensors (4)	$22.56	Creatron Inc
Standoffs F/F	$3.38	Sayal Elec
Portable Battery	$45.19	Scosche
Philips Head Screws	$5.64	Sayal Elec
Phase 1 Total	**<$200.00**	
Phase 2		
Materials to improve functionality, fit, and finish of project.		
Phase 2 Total	**TBD**	
Phase 3		
Off campus colocation	<$100.00	An example: [4].
Shipping	*TBD*	
Tax	*TBD*	
Duty	*TBD*	
Phase 3 Total	**TBD**	

Concluding remarks

This proposal presents a plan for providing an IoT solution for AutoVehicle. This is an opportunity to integrate the knowledge and skills developed in our program to create a collaborative IoT capstone project.

Abstract

So far human controlled robots have been great; they are able to complete many tasks instructed by the user that's controlling them. As technology advances in our world we have learned that a new form of robotic programming has been developed to allow the robot to complete tasks on their own, independent of the user. Artificial Intelligence(AI) has enabled developers to create robots that are capable of operating on their own, making them even more useful and easier to operate. To have a robot that is able to independently map an unknown area would be great in many cases. One such case would be: if an area that a user wants to map is small or has a small entrance where they may not be able to physically enter, it is useful to have a small robot that can complete that task on its own and return a visual map of that area to the user. The Auto Vehicle has the ability to autonomously map an unknown area with the press of a button on the mobile application. It independently avoids obstacles in its path while always having the knowledge of its position in the unknown area. The vehicle will record the data in the form of coordinates to be sent to a database which can be retrieved by the mobile & web applications. The coordinates, when retrieved by the mobile application, will then be displayed in the form of a 2D map of the area.

Table of Contents

List of Illustrations

Introduction

There are numerous areas such as: mines and sewer systems which are beyond the reach of humans. These areas may either be too rigid or have entrances that may be too small for an average sized person to reach. This report describes the process followed for the design and utilization of an autonomous vehicle with the ability to map areas that may represent an inconvenience for humans to enter. The autonomous vehicle in question will be fully autonomous and will use ultrasonic sensors to navigate and record coordinates in an area which will be sent to a database. This database can then be retrieved by website to display them to specific users when they log in. The mobile application will retrieve the coordinates from the database in addition to a drawn map of the area which it will be able to display to the user. The application will also be able to remotely control the car through the usage of an onscreen joystick, through a Bluetooth or WiFi connection. The information presented will be centered on the design of the autonomous vehicle in addition to the mapping algorithm which it will implement.

System Requirements Document

1. Introduction

1.1 Purpose

The purpose of this small scale vehicle is to be able to autonomously navigate and map an area. Autonomous vehicles are the current trend in technology and this vehicle will allow a user to make a map of an area without having to physically carry out the task.

1.2 Definitions

An autonomous vehicle is a vehicle that is capable of sensing its environment and navigation without human input(Gehrig & Stein, 1999)

1.3 Overview

This vehicle is made using a combination of Raspberry Pi 3, Ultrasonic sensors, and a motor driver. These components are built onto smart car chassis which will be used to move. The ultrasonic sensors are used to detect obstacles and send the information back to the raspberry pi.

1.4 Target Audience

The target audience of this project is students with a background in Computer Engineering Technology who may have not taken part in a project of this type before. These students students will typically be in their final year of study.

2. Overall Description

This section will give an overview of the whole system. The system will be explained to show the different components of the system interact with one another and introduce the basic functionality of it. It will describe what type of users will be expected to use the system and what functionality will be available to them. In addition to that, the constraints and assumptions for the system will be outlined.

2.1 Work Breakdown

Introduction

The hardware and mobile application were a collaborative effort during the first stage of the project, but as of the second stage this is how the work was divided.

Hardware

The Auto Vehicle's chassis has all the hardware connected or mounted on it that operates together to achieve the functionalities that are required from it. The most important piece of hardware is the Raspberry PI 3, the microcomputer which has all the other hardware connected to it. This is where it receives data from and send data t. It also makes it possible to connect to the mobile application through a WiFi or Bluetooth connection. The hardware that communicates with the microcomputer are the Ultrasonic Sensors which are distance sensors, they are used to detect objects that may appear in front of the vehicle then send a signal to the microcomputer where it process it then manipulates the actuators according to the data that it received, the actuators are DC Motors used to move the vehicle, but before that the microcomputer actually sends data to the H-Bridge the driver which based on the data controls the DC Motors direction.

(Developed by Jan Yalda)

Database

The system requires two databases, one will be used to store the users that have signed up to use the mobile application and another to store data of the mapped area. The mobile application user's database is a MySQL database on a free hosting remote server, its main purpose is to store the information of users that sign up for the application. This information consists a username and an encrypted password which will allow the user enter the mobile application. The second database, which is still in development, will be used to receive data from the hardware while it is mapping an area. This data will be stored in the database which can be retrieved by the mobile application to display the mapped area to the user.

(Developed by Jan Yalda)

Web Interface

A website will be developed to allow the user can login to his account. Each specific user will be able to see their previous data been collected using the AutoVehicle. One section of the web site will have a

small biography for the developers of the project and their contributions. Another page will contain a step by step a copy of this technical report. In addition to those, a section will be added to allow visitors give us feedback and suggestions on how we may improve the project.

(Developed by Bilal Al-fanous)

Mobile Application

The AutoVehicle Application is used to communicate with the autonomous vehicle remotely. It has in total 13 java classes which includes 6 activities. The functions of the activities are as follows: Login Activity is the first page the user will see when they open the application, it is where the user enters their information to get their user specific information which is stored in the database; MenuActivity is the page the user sees after they successfully login, they can now choose whether they want to control the car or retrieve a map; The register activity is where new users sign up with their information to be stored in the database. After registering users will be taken back to the login page; The first remote control activity has a joystick which the user can use to control the car over an internet connection; The remote control(BT) activity is similar to the first remote control activity except it allows the user to control the can over a bluetooth connection; the settings activity is where the user can enter the ip address of their car so that they can connect to it; the mapping activity is where the user can see tell the car to start mapping and retrieve the map that their car has made. The application will work in unison with both the database and hardware to be fully functional. In addition to these various features, the application also has support for both English and French.

(Developed by Khemar Bryan)

2.2 Product perspective

The system will consists of 3 parts: one mobile application, one 4 wheeled robot(car) and one database. The mobile application will communicate with the car and control it using an onscreen controller, it will also be able to retrieve a map from the car and send it to the database. The database will allow users to log into the application and store user related information.

The 4 Wheeled car is a 4 wheel drive robot smart car chassis that has various components mounted onto it. It need to be able to operate on its own and run python and C programs. To do this it will use a Raspberry Pi 3 modules. It will also need to detect obstacles and move its wheels using the motors. These functions will be controlled by 2 ultrasonic sensors and a H Drive. All these components connected as a whole will allow the car to be autonomous.

The mobile application will need to communicate with the car over a Bluetooth connection using a built in Bluetooth Adapter on both devices and it will communicate with the database through an internet connection. The Bluetooth and internet functionality will be written into the the application in order to allow the user to use these functions seamlessly.

The mobile application has login functionality which requires a user be existing in the online database. The mobile application sends a request to the database to check if the user credentials provided matches one of the users that exists in the database then sends a response back to the mobile application which indicates if the user exists or not and therefore gives or denies access to the main functionalities of the application. If the user does not have a login they can signup for one.

After a successful login to the application has been made the two main functionalities are presented. First the functionality to use the Auto Vehicle as a remote controlled car, with this functionality the mobile application communicates with the hardware using the internet and sends commands to be received by the hardware and acted upon which are just basic remote control functionalities. The second functionality the mapping of an area communicates with the hardware the same way, it just orders the hardware to do the mapping functionality instead.

The database will be hosted on an online server and it will be store data from the application. As stated before, the mobile application will require access to the database to send login information and retrieve a response to allow the user to access it's functions.

There are some restrictions as it relates to connectivity on both the hardware and the mobile application. Although the application can communicate with the hardware over a WiFi network it is limited to certain WiFi networks. Enterprise WiFi, for example will cause the application to crash when trying to connect to the car. To avoid this problem Bluetooth functionality will be available as an alternative.

2.3 Product functions

Using the mobile application, users logged in will have two main choices on a menu first the remote control functionality and second the mapping functionality. When the users choose one of them they

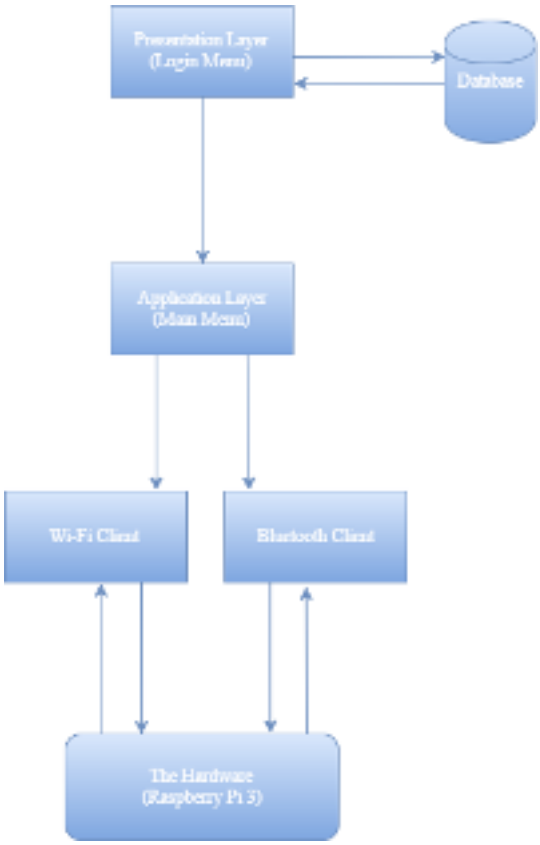

Figure 1:

will be taken to the activity that is made for the functionality, where if its the users first time logging in gets an alert saying that an IP address is required which is the IP address of the Auto Vehicle, the user can just to add it right then or can ignore the alert and provide it another time.

When an IP address is provided users going back to the two activities will have the ability to actually connect to the Auto Vehicle and start any functionality that they wish to perform.

The remote control activity provides the users with the option to connect to the hardware using the connect button and start controlling it using the virtual joy stick provided in the remote control activity and finally when they no longer want to continue with this functionality they have the choice to disconnect from the hardware using the button provided.

The mapping activity has the connect and disconnect buttons that have the same functionality as in the remote control activity, but it also has two other buttons one to start the mapping process and the other to stop it, when the stop button is click the mapped will be send from the hardware to the mobile application and displayed in this activity.

2.4 User characteristics

There will be two types of users of this project, the owner of the project and users of the mobile application and hardware. Both of these users will have different uses of the system and therefore will have different

requirements.

The owner of this project will have access to the database which allows them to make changes to it as they please. They will have their own login credentials have access to the mobile application and hardware.

Users of the mobile application and hardware will only have access to those two components. This means that the users will require login credentials to use the application. They will be allowed to signup and login to the application which would allow them to access it's functionalities. By proxy of the application they will have access to the hardware as well, through the connection between the hardware and application.

2.5 Constraints

The mobile application is constrained by the internet connection. Since the application sends to and receives data from the database over the internet it is crucial that there is an internet connection.The application also requires there be an existing internet connection to connect to the car over WiFi.

The system UI may provide another constraint on the mobile application. The application requires that the phone be running on at least Android API level 19, to be downloaded.

Bluetooth connectivity is also a constraint on the mobile application. It can connect to the car through Bluetooth but it requires that both the raspberry pi and the device the application is running on have Bluetooth adapters. Using Bluetooth to connect also cause provides a proximity issue. The connection requires that the receiver receiver and sender both be within a certain range of each other to be able to connect efficiently or at all.

Another constraint is that the mobile application is only available in two languages: English and French. Users will be required to understand one of those two languages to use the application effectively although it is not impossible.

The car requires that external libraries be installed onto the raspberrypi so that it can access the Bluetooth functionalities of the program running on it. If these libraries are not installed the program on the car will not be able to execute.

2.6 Assumptions and Dependencies

An assumption of the system is that it the users to will have an android device that runs on Android API level 19 and above as a minimum requirement to use the mobile application on their smart phones. Users with devices that don't meet the requirements will not be able to download and use the mobile application.

Another assumption of the system is that users will have devices with operational Bluetooth and WiFi adapters.

If the user does not have enough space on their device they will not be able to download and use it. It is assumed that users will have at least 15MB of space free on their device.

3. Specific Requirements

This section contains all the of the functional and quality requirements of the system. It gives a detailed description of the system and all its features

3.1 External interface Requirements

This sections provides a detailed description of all inputs and outputs into the system. It also gives a description of the hardware and software involved in the system.

3.1.1 User Interface

When a user opens the application for the first time they should be greeted by a login page. Assuming the user does not have login credentials, they can signup by going to the signup page which is an option in the login page. After signing up the user can go back to the login page. For future instances users have the option of the application remembering their username so that when they open the application the next time they only have to enter their password.

After logging into the application the user will be taken to the main menu where they can choose to use any of the applications main functionalities, which are: the remote control(over WiFi), remote control(over Bluetooth) and mapping.

When users attempts use the first remote control function they will be prompted to go to settings and enter an ip address. This ip address is required for the user to connect to the robot over an internet connection. After entering the ip address users can go back to the remote control app and select the connect option. After connecting they will now be able to move the car with the remote control and a direction will be displayed corresponding with the position of the joystick.

The mapping activity also prompts the user with an onscreen prompt to go to settings. When the user goes to settings they can turn the Bluetooth on with the touch of a button. After turning the Bluetooth on the user will now be able to use the mapping functionality. The user must tap the start button and after doing so they will receive an onscreen notification saying they have connected. When the user taps stop they be notified that the map has been received and it will be displayed.

The remote control with Bluetooth activity is very similar in functionality to the remote control with WiFi activity. The main difference is that in this activity the user will connect to the car through a Bluetooth connection rather than an internet connection. If the user does not have Bluetooth turned out or their device does not have a Bluetooth adapter they will not be able to use this functionality.

3.1.2 Hardware Interface

The designated hardware for this project is a Raspberry Pi 3 module that is mounted on a 4WD chassis. This module is the main component of the car that all the other physical components are connected to and communicate with.

Raspberry Pi 3

The Raspberry Pi is the main and most important component of the hardware. This is where all code will be stored, compiled and run from from. This was the only pre-specified piece of hardware as it was a strict requirement for this academic project.

Overview and Specifications:

- Dual step-down (buck) power supply for 3.3V and 1.8V
- 5V supply has polarity protection, 2A fuse and hot-swap protection
- USB/Ethernet controller chip
- 4 USB ports
- 40 GPIO pins. The top/first 26 pins match the original layout of old raspberry PIs, 9 additional GPIO and 2 EEPROM Plate identification pins
- Composite (NTSC/PAL) video integrated into 4-pole 3.5mm 'headphone' jack
- MicroSD card socket
- Four mounting holes in rectangular layout
- Size: 85mm x 56mm
- Processor, Broadcom SoC running at 700MHz (can be overclocked)
- RAM, 512MB soldered on top of the Broadcom chip
- power connector, microUSB
- software – riparian will be used as an OS
- HDMI port
- Audio part of the A/V jack is the same
- Camera and DSI Display connector

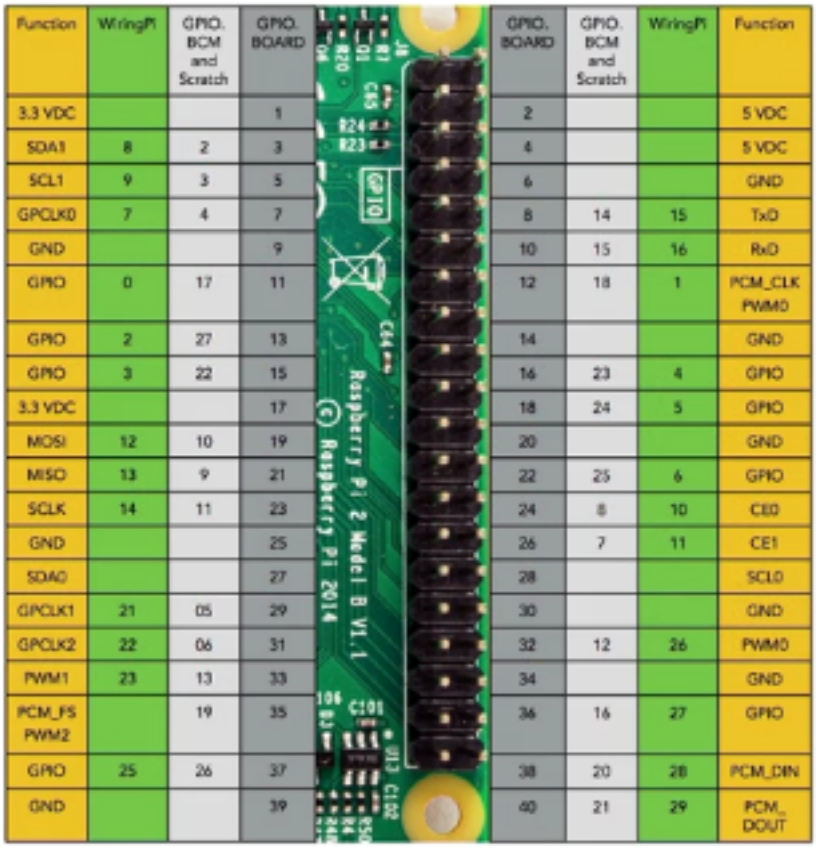

Figure 2:

H Bride L298N Motor Driver

The first piece of hardware that the Raspberry Pi 3 connects to is the L298 H Bridge. The H bridge is used to control the motors speed and direction. It is also used to provide extra voltage in the case of the Raspberry Pi not being able to provide enough due to the fact that it has another source of power. It works by allowing a voltage to be applied across a load in either direction.

Pins:

- Out 1: Motor A lead out
- Out 2: Motor A lead out
- Out 3: Motor B lead out
- Out 4: Mo (*Can actually be from 5v-35v, just marked as 12v*)
- GND: Ground
- 5v: 5v input (*unnecessary if your power source is 7v-35v, if the power source is 7v-35v then it can act as a 5v out*)
- EnA: Enables PWM signal for Motor A (Please see the ”Arduino Sketch Considerations” section)

- In1: Enable Motor A
- In2: Enable Motor A
- In3: Enable Motor B
- In4: Enable Motor B
- EnB: Enables PWM signal for Motor B

Specifications:

- Double H bridge Drive Chip: *L298N*
- Logical voltage: *5V Drive voltage: 5V-35V*
- Logical current: *0-36mA Drive current: 2A (MAX single bridge)*
- Max power: *25W*
- Dimensions: *43 x 43 x 26mm*
- Weight: *26g*

HC-SR04 Ultrasonic Ranging Module

There are two of these modules and they are used to detect objects within a certain range of the car. The ranges can be toggled using code on the raspberry pi but the specified range is between 20cm - 400cm. It is accurate up to 3 mm. The module has ultrasonic transmitters, receiver and control circuit.

Electric Parameters:

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
Measuring Angle	15 degrees
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

Hwydo 4WD Robot Smart Car Chassis

This chassis was chosen because it was inexpensive and it met our requirements. For this project we wanted something that the raspberry pi could fit on top of, able to balance itself and turn efficiently. We

briefly considered a 3 wheeled chassis which was able to balance but the turning was not as efficient.

Specifications

- Size: 27 x 14 x 4 cm (Slightly larger than our required size)
- Carrying Capacity: 1 KG
- Gear Motor rotate speed: 125 rev/min
- Gear Motor Power: 6 V
- Weight: 600 G
- Powered by: 6 AA Batteries

Included in the chassis is:

- 2 x Acrylic Plates
- 4 x DC Gear Motor (125/min,6V)
- 4 x wheels
- 2 x encoder disk
- 1 x AA x 4 Battery Holder
- 1 x Power Switch
- Full set of screws and nuts

Dual Shaft Gear DC Motor

This component is included in the the smart car chassis. This gear box is applied for tracing car or robot.

Technical Details:

Product Name	DC Geared Motor
Rated Voltage	DC 3V - 5V
Rated Current	~160mA (at no load)
Speed	< 120 RPM at No Load
Plastic Tire Wheel Size	65 x 25mm/ 2.6″ x 0.98″ (D*T)
Motor Size	65 x 22 x19mm/ 2.6″ x 0.87″ x 0.75″ (L*W*T)
Color	Yellow, Black
Material	Metal, Plastic
Weight	67g

3.2 Functional Requirements

This section includes the requirements that specify all fundamental actions of the system.

Functional Requirement 3.2.1

Title: Registration

Description: Users should be ale to register through the mobile application. The user must supply a fullname, username and password to be entered into the database.

Functional Requirement 3.2.2

Title: Login

Description: If a user has already registered for the application they should be able to login.

Functional Requirement 3.2.3

Title: Main Menu

Description: Given that a user has logged into the application they should be able to choose which function they want to utilize

Functional Requirement 3.2.4

Title: Toggle Bluetooth

Description: Given that users are logged into the application they should be able to toggle Bluetooth on and off without having to use the device's UI.

Functional Requirement 3.2.5

Title: Connect to car

Description: Users should be able to connect to the Raspberry Pi mounted car through Bluetooth or over an internet connection

- An internet connection can be established by entering the IP address in settings and connecting from the first remote control activity.
- A Bluetooth connection can be established in both the Mapping and Remote Control(BT) Activities by toggling on Bluetooth in settings and connecting through either activity
- Both these functions require that the programs be running on the Raspberry Pi

Functional Requirement 3.2.6

Title: Control Car

Description: Given that users have connected to the car through either Bluetooth or WiFi they should be able to move the car using an onscreen joystick or starting the the autonomous feature.

Functional Requirement 3.2.7

Title: Display a map

Description: Given that a Bluetooth connection has been established users should be able to start the autonomous feature of the car and display a map of the area when the stop button is tapped

Functional Requirement 3.2.9

Title: Support French

Description: Given that a user's device language settings are set to French, the app should be able to support this language

Functional Requirement 3.2.10

Title: Login - Website

Description: Given that a user's information is present in the user database, the user should be able to login on the website.

3.3 Performance Requirements

3.3.1 Usage of the Remote Control

The remote control should be easy to use and give the user feedback. In addition, whichever direction they move the joystick in the car should move in that direction.

3.3.2 Usage of features

The different features should be evident and in plain view of the user

3.3.3 System Dependability

If the application is unable to connect to the car the user should receive some sort of feedback

3.4 Design Constraints

3.4.1 API Level

The API level the app is designed to run on.

The application requires that the user's device be running on at least Android API level 19

3.4.2 Device Internal Storage Space

The amount of space the application requires.

The application requires at least 14 MB of internal storage

3.5 Software System Attributes

3.5.1 Bluetooth Connection

The application should be able to connect through Bluetooth in order to communicate with the car.

3.5.2 Internet Connection

The application should be connected to the internet in order to communicate with the database.

3.5.3 Application Extendability

The application should be written in a way that favors modification.

4. Build Instructions

4.1 Introduction

This content of this section will give detailed instructions on how to recreate the project.

4.2 Time Commitment

The table below gives an outline of the estimated time required to complete this project. It should approximately take an entire day to complete if you follow all the steps detailed in this document.

Task	*Time Estimate*
Acquire parts	1 day
Assemble chassis	30 minutes
Print PCB(Optional)	2 hours
Solder components onto PCB (if using PCB)	1 hour
Setup sensors on breadboard (Optional if not using PCB)	10 minutes
Setup all wiring and connections to the Raspberry Pi	45 minutes
Configure Rasperry Pi with OS	1 hour
Setup Android Application	40 minutes
Setup Code for on raspberry p for car operation	20 minutes
Testing	1-2 hours

4.3 Budget

These prices may vary depending on which supplier you purchase them from. All these parts excluding the Raspberry Pi can be purchased from Creatron Inc. The raspberry pi can be ordered online through amazon. Parts with a "*" can be replaced with suitable alternatives.

Item	*Cost(Before Tax)*	*Tax*	*Cost(After Tax)*
Raspberry Pi Kit	$ 99.99	$ 13.00	$ 112.99
Hwydo 4WD Robot Smart Car Chassis*	$ 38.00	$ 4.94	$ 42.94
Energizer Max AA Batteries*	$ 4.97	$ 0.65	$ 5.62
HC-SR04 Ultrasonic Sensors (x2)	$ 4.99	$ 0.65	$ 5.64
6" Male to Male Jumper Wires (x10)	$ 2.97	$ 0.39	$ 3.36
6" Male to Female Jumper Wires (x10)	$ 2.98	$ 0.39	$ 3.37
6" Female to Female Jumper Wires (x10)	$ 2.99	$ 0.39	$ 3.38
L298 Dual Motor Driver-2A	$ 14.00	$ 1.82	$ 15.82

Total	$ 170.89	$ 193.11

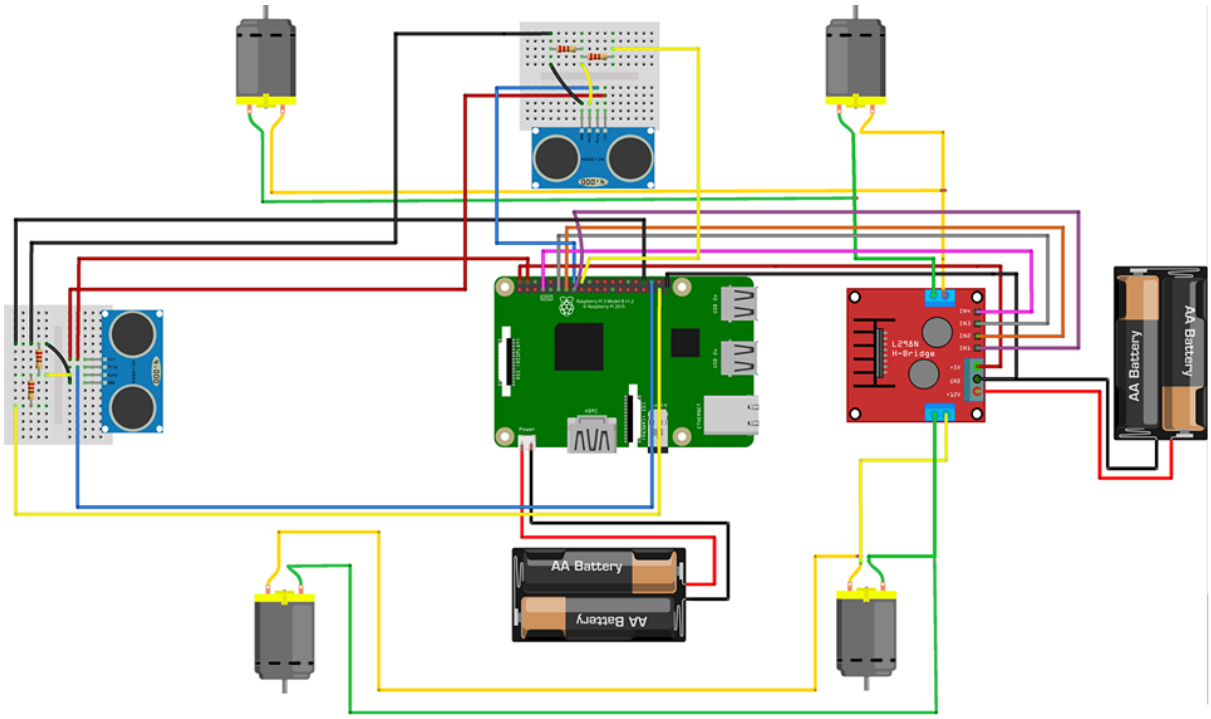

Figure 3:

4.4 System Diagram

Figure 3 illustrates how all of the components on the AutoVehicle are interconnected. The raspberry pi is the central hub for all the devices to communicate within the structure.

4.5 Mechanical Assembly

To start unbox all the parts that come in the 4WD chassis kit then start by:

- Putting the motor mounts on the chassis
- Putting the actual motors on to the mounts
- Putting the standoffs on the bottom chassis layer
- Then screwing the top layer to the standoffs
- Finally attach all the tires to the motors
- Then find good spots on the chassis to mount the raspberry pi, sensor circuit which is on a bread board and the H-driver
- Place the H-driver in a position close to the motors
- To connect the H-Driver to the motors, follow these steps:

  - Loosen the screws on the OUT ports (OUT1, OUT2, OUT3, OUT4)
  - Put the red and black leads from the right motors of the car into OUT2 and OUT1 respectively then tighten the screws.
  - Place the red and black wires from the left side into OUT4 and OUT3 respectively then tighten the screws.

- After the H-driver is connected to the motors you must now connect the driver to the raspberry pi and the power supply
  - Loosen the screws on the 5V, round and VCC ports
  - Take the red lead from the power supply and place it into the VCC port then take the black lead and push it into the ground port. Tighten the screws on the VCC port but leave the ground port open for now
  - a male to female wire and plug the header onto pin 2 on your raspberry pi (this is the 5V out pin) then push the male head into the 5V port on the motor driver.
  - Using another male to female wire, plug the header onto pin 6 on the raspberry pi and push the male head into the ground port on the motor driver.
  - You can now tighten the screws on the 5V and ground ports
- And finally place the sensor circuit somewhere on the edge of the chassis so nothing will be interfering with the sensors like the wires.

4.6 PCB/Soldering

The files needed to recreate the PCB can be found at this link: PCB layout

Soldering onto the PCB is a simple task that even beginners can carry out but it is recommended that you have at least basic knowledge of circuitry. Practice before you solder onto your pcb to minimize mistakes

Step 1

Get a soldering iron, a PCB holder and some solder.

Step 2

Place the PCB into the PCB holder and plug the soldering iron in to heat it up

Step 3

When the LED on the soldering iron has turned green this means you can begin soldering

Start by cleaning the putting some solder onto the iron and wipe it off on a sponge (this is done to clean the iron)

Step 4

Look at your schematic then begin soldering on the necessary parts. (See figure 4)

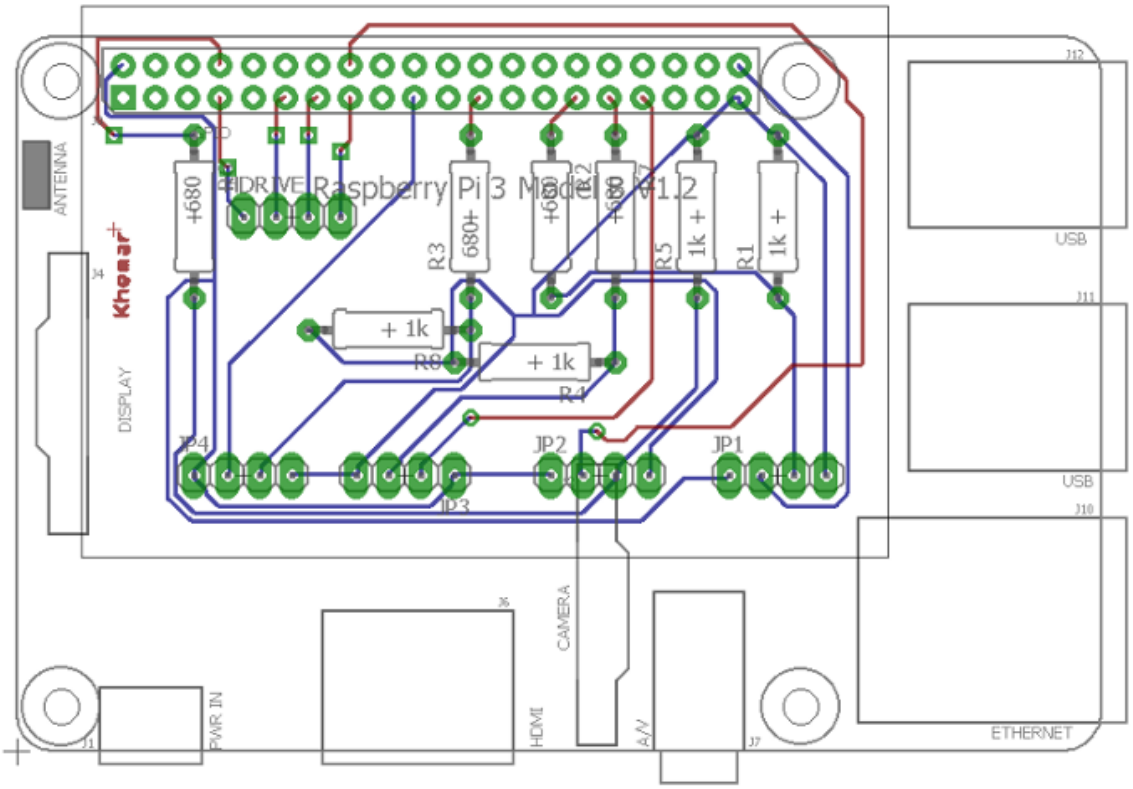

Figure 4:

4.7 Power Up

In this phase, you will be powering up your car and PCB(if you chose to use one)

Powering up the PCB

- The raspberrypi by default comes loaded with an OS but it is recommended you reinstall it
- Format the SD card then use http://sourceforge.net/projects/win32diskimager/ to write the image file https://downloads.raspberrypi.org/raspbian/images/raspbian-2016-09-28/2016-09-23-raspbian-jessie.zip onto it.
- Using the HDMI cable from the rapsberrypi kit connect the pi to a monitor
- Using an Ethernet cable connect the pi to a source of internet.(or use wifi)
- Put the sd card into your raspberrypi and power it on.
- When the raspberry pi is powered on look for the ip address (hover cursor over internet icon or use ifconifg in the terminal)
- Open remote pc, enter the ip address and connect to the pi.
- Go to the six0four.github.io repository and look in the ModularSenseHatStripped folder for a file called traffic 2B.c. Download the file to your desktop.

- Put the PCB header onto the raspberry pi's pins
- Go to the terminal and run the traffic 2B.c program, the LED's on the PCB should now start changing colours.

Powering up the Car

- Put the batteries into the battery holder
- If the H Drive is receiving power from the batteries the LED should illuminate
- Remove one of the batteries from the battery holder to turn of the motors as you will need to test if the raspberry pi is powering the motor driver
- Plug in the raspberry pi and the LED on the motor driver should illuminate once again
- Now put the batteries back into the battery holder and all components should be powered up

4.8 Additional Installations & Modifications

The code written for utilizing the Bluetooth library is written in Python. Although Python has a large and comprehensive library, it does not yet have a standard library for Bluetooth. As such, to utilize the Bluetooth functionality for this project there are additional libraries that need to be install. In addition to those libraries, there is also a file that needs to be modified. These will be explained further as follows.

PyBluez

PyBluez is an extension module written in C that provides access to the rasperry pi's Bluetooth resources. To install it you simply need to log into your raspberry pi and open the terminal window. After this all you need to do is run the command:

```
sudo apt-get install py-bluez
```

SDPTool

SDPTool is a library in linux that provides the interface for performing SDP queries on Bluetooth devices. To utilize this code you will need to load the serial port profile by using the command

```
sudo sdptool add SP
```

In addition to running that command you will need to run the Bluetooth daemon in compatibility mode. You can do this by editing the /etc/systemd/system/dbus-org.bluez.service file. Instructions to do this are as follows:

- Find the location of bluetooth.service by

```
systemctl status bluetooth.service
```

- After find the location go to that location edit bluetooth.service
- Look for

```
ExecStart=/usr/libexec/bluetooth/bluetoothd
```

- add -C or --compat at the end of this line and save it.
- Run the command

```
service bluetooth start
```

- Finally reset the adapter using

```
sudo hciconfig -a hci0 reset
```

After making these modifications you should be able to run the Bluetooth python code.

4.9 Website Designing

4.10 Mobile Application

The files needed to recreate the android application can be found at the following link: https://github.com/bilfnous/AutonomousVehicle

You will need a physical device that runs on at least Android APK level 19(4.4 & above) to run this application. A virtual android device is not recommended as it does not have access to bluetooth.

- Follow the link above to go to the repository where the android code is stored.
- Download the repository or follow this direct link https://github.com/bilfnous/AutonomousVehicle/archive/master.zip
- Unzip the folder
- Open Android Studio
- Go to file, new, import project, locate the project you just downloaded and click OK
- After importing the project, wait for gradle to load then make sure that all the java classes from the repository can be found there.
- Before continuing, enable developer options on your android device by going to settings, software info

- Tap build number multiple times until a pop up appears saying you are now a developer
- Go back to the settings menu and enter developer options, from there turn on developer options and enable USB debugging
- Plug your device into your computer
- Run the project in android studio by click the green triangular button or use the run menu.
- After clicking the run button you will be met with a screen that asks you to choose which device you want to run the application on, choose your device

4.11 Database

The Database holds the users login information along with each users retrieved map send from the Hardware(Raspberry Pi) to be send and displayed on the website. its structures consists of 5 fields; user_id which an auto incremented Integer which gives each user an id, a name which is a Varchar that holds only the users first name, a username which is a Varchar that hols the users username, a password which is also a Varchar which holds the hashed password of each user and finally map a Longtext field which holds the map. To recreate the database database you must:

- Signup for an account on 000webhost.com
- Navigate to Database then click mysql to create a database
- Enter information in the specified fields then click create database
- Wait a few minutes for the database to be created and loaded then click phpmyadmin and select the database you created by clicking enter phpmyadmin
- From here you will create a table with the specified fields
- Download the files Login.php and Register.php from our repository
- Scroll down to file manager and click on filemanager 2
- Enter the public-html folder and place the files you just downloaded into this folder

5. Project Schedule

Hardware Timeline

Week #	Task	Date
Week 1	Selecting Project	09/06/2016 - 09/12/2016
Week 2	Project Proposal	09/12/2016 - 09/19/2016
Week 3	Meeting	09/19/2016 - 09/26/2016
Week 4	Total Budget	09/26/2016 - 10/03/2016
Week 5	Components Purchased	10/03/2016 - 10/10/2016
Week 6	Assembling Components	10/10/2016 - 10/17/2016
Week 7	PCB	10/17/2016 - 10/24/2016
Week 8	Placard	10/24/2016 - 10/31/2016
Week 9	Progress Report	10/31/2016 - 11/07/2016
Week 10	Peer Grading Demonstration	11/07/2016 - 11/14/2016
Week 11	Individual Build Video	11/14/2016 - 11/21/2016
Week 12	Progress Report	11/21/2016 - 11/28/2016
Week 13	Presentation	11/28/2016 - 12/05/2016
Week 14	Build Instructions	12/05/2016 - 12/12/2016
Week 15	Final Script	12/12/2016 - 12/19/2016

Integration Timeline

Week #	Task	Date
Week 1	Scheduling Section	01/09/2017 - 01/13/2017
Week 2	Project Status	01/13/2017 - 01/20/2017
Week 3	Software Requirements	01/20/2017 - 01/27/2017
Week 4	Status Update	01/27/2017 - 02/03/2017
Week 5	Status Update 2	02/03/2017 - 02/10/2017
Week 6	App, Web, and Database Individual Demonstration	02/10/2017 - 02/24/2017
Week 7	Status Update 3	02/24/2017 - 03/03/2017
Week 8	Group Integration	03/03/2017 - 03/10/2017
Week 9	Group Troubleshooting	03/10/2017 - 03/17/2017
Week 10	Status Update 4	03/17/2017 - 03/24/2017
Week 11	Final Report	03/24/2017 - 03/31/2017
Week 12	Status Update 5	03/31/2017 - 04/07/2017
Week 13	Project Demonstration	04/07/2017 - 04/14/2017
Week 14	Group Presentations	04/14/2017 - 04/21/2017
Week 15	Final Video Script	04/21/2017 - 04/28/2017

6. Progress Reports

Throughout this project we were required to provide our professors with reports on our project throughout the school year. In the first semester this project were split into two different classes focusing on hardware and software respectively. In the second semester there was one class focused on the integration if the hardware and software projects. This section will give show our progress throughout the cycle of this project and the school year.

Phase 1 Progress Reports

Week 2 In this week our project team was formed and we drafted our first proposal for AutoVehicle.

Week 3 Project Schedule was created and submitted

Week 4 Researched all the necessary components and crafted a budget.

Week 5 Purchase of all necessary components

Week 6 Mechanical assembly milestone, the chassis was assembled

Week 7 The PCB was soldered and demonstrated

Week 8 Created a placard in preparation for open house

Week 9 Recorded and presented Build Video

Week 10 Demonstration of the hardware

Week 11 Individual build log entries

Week 12 Production testing

Week 13 Individual presentations

Week 14 Build instructions were uploaded

Week 15 Recorded video giving brief details about the project

Integration Status Reports

February 3, 2017

Status Summary

With the remote-control part of the project completed we are now focused on the mapping functionality and making adjustments to improve the hardware and the software sides of the project. The android application is being modified to prevent crashing when an incorrect IP address is entered and ways of implementing the mapping functionality is being viewed to have a better understanding of how it works.

Project Overview

The mapping functionality is still in the research phase.

The android application is being modified to prevent crashing and incorporate Bluetooth connectivity.

The web site is under development.

Problems and Opportunities

Mapping is a very hard concept to comprehend and implement. We've gathered information from various sources which mainly include complex algorithms on how to do it but we're not quite sure how we can use it. This provides us with the opportunity to collaborate with someone who has experience with robotic mapping.

During the last semester, we used a WiFi/Internet connection to communicate with the car. This worked well from our homes but connecting to an enterprise WiFi proved to be an issue. This has provided us the opportunity to learn about Bluetooth connectivity and how we can incorporate that into our project.

Budget Overview

No changes to the budget thus far, we haven't had to add anything to the project.

Conclusions

Overall the project is still in development. Our goals for this month are to prevent the application from crashing, understand and implement the mapping functionality and have a skeleton for the website.

February 17, 2017

Status Summary

This week we met with our collaborator to discuss our options, as it relates to mapping. After meeting with the collaborator new solutions were introduced, and his ideas were taken into consideration. The rest of our time was focused on enabling connectivity on our car. A solution has been found to connect the vehicles to enterprise Wi-Fi. Project website started to take its overall design and outlines.

Project Overview

The mapping functionality is still in the research phase but will now see some progress after we take into consideration the options brought to us by our collaborator. The Bluetooth feature for now will be on hold, as we've found a solution for the Wi-Fi connectivity. The main page of the website is being coded and the final outline of the website is taking shape.

Problems and opportunities

Jan: A meeting with a collaborator in the mapping field took place, and some very interesting ideas were introduced. An electronic peripheral might be used, such as IMU to enable us to sense angles. Another

solution known as the flood fill algorithm, is also being considered along with other ideas to do door mapping.

Khemar & Jan: Bluetooth proved to be a strenuous task and it will be on hold for the time being. While working on the Bluetooth functionality the Wi-Fi connectivity aspect was fixed in the application. In addition to that, a solution to our enterprise Wi-Fi issue was found by Jan. Moving forward, Wi-Fi will be our main way of connectivity until we learn more about Bluetooth.

Bilal: Website development has been going smoothly up to now, and there is not any major problem that worth mentioning, the major development was on the skeleton and overall design of the website.

Link for AutoVehicle's website:

www.munro.humber.ca/~n00994056/index.html

Budget Overview

After meeting with the mapping collaborator, we may have to adjust our budget. Among other things, we have taken into consideration an IMU, which should cost around 50$.

Conclusions

Overall, there has been major progress in regards to the connectivity aspect of the project, our goals have been met for this month with the app no longer crashing when and invalid IP address is entered. We now have a clear idea of the options we have to aid us in creating mapping functionality. The general outline of the website has been shaped and for the next month the website will continue to be in development, some initial mapping code and testing should take place.

March 10, 2017

Integration Progress

After coming to the conclusion that the Wi-Fi is not the best option to go with because of restrictions that Humber Wi-Fi has, we have decided to use Bluetooth as the main method of communication between the Raspberry Pi and the Android application. Our integration is not yet complete although we have seen progress with the application and hardware. We now have a connection between the Raspberry Pi and the Android application in addition to the android application and the database. The only connection left to establish is between the website and database.

Project Overview

The mapping functionality is in progress. So far, a visual of the mapped area can be viewed from the raspberry pi when the program is run. Jan will work on sending the map to the Android application through Bluetooth.
The android application now has the ability to connect to the raspberry pi using Bluetooth and control the basic functions of the car. The main goal for Khemar moving forward is to work on retrieving the mapped area and displaying it.

Bilal is working on establishing a connection between the website and the database.

Problems and Opportunities

Our main problem currently is establishing connection to the database using the website, so the user can sign up/login and view the mapped area.

Budget Overview

No changes to the budget thus far.

Conclusions

Overall the project is going well after resolving the connection issue by using Bluetooth. We now need to work on sending the map to the Android application and displaying it, also the connection between the website and the database needs to be established.

March 14, 2017

Troubleshooting status

Throughout the duration of this project there have been numerous hindrances that prevented the integration of the car and the mobile application. From connectivity issues on the application and vehicle, to problems understanding how to implement particular programs such as mapping. This will be an outline of some of the troubleshooting we've done throughout the project.

Wi-Fi connectivity on the application and hardware sides were issues that plagued us for most of the project. Firstly, we were unable to connect to enterprise Wi-Fi and this was a problem because the application was reliant on a server running on the car. Whenever the application attempted to connect to the server on campus it would crash because the server was offline. We resolved the enterprise Wi-Fi problem by specifying the location of the Humber certificate when writing the network information in etc/wpa_supplicant/wpa_supplicant.conf and placing the actual certificate downloaded from Humber's IT website in /etc/certs. After doing that we configured a static IP address on the car and it allowed us to connect once but after that it wasn't possible due to how the Humber's network is configured. These solutions only made our problem worse by giving us more issues to resolve, one of which was the VNC connection. In the end, we decided it was best to format the SD card and reinstall the Raspbian OS to solve the issue. The overall connectivity issue however was solved by going a completely different route and opting to connect the application and car over Bluetooth instead.

Mapping functionality was a really complicated part of the project when researching methods of implementing it. The main reason was that we had most of our code for the hardware part in python a language that isn't that hard to learn, but it's better to just go with a language we are already familiar with, mainly to allow us to use our time more efficiently to work on the actual mapping functionality. This issue was resolved by converting the code we have written in python for the hardware into the C language. This helped us complete the mapping functionality faster.

Issues were also faced while building the website. Finding somewhere to host the website proved difficult for a number of reasons. Firstly, a google service was considered, but because the data base we are trying to connect is json and not SQL, the idea of hosting the website on fire base was forfeited. AWS

was the next consideration, and I have been reading some tutorials on amazon's website on how to set up a server and how to host a website on it. For the time being the hosting issue has been solved by using the 000webhost website to host the site. It is now possible to login to the website. In the meantime, I'll also keep working on AWS. Once I have figured out everything on AWS my plan is to possibly host the site there since AWS is commonly used by developers.

7. Testing

7.1 Unit Testing

Testing the H Drive & Battery Power Up

Place the batteries into battery holder and the LED should turn on and shine red

Testing H Drive & Raspberry Pi Power Up

Plug the raspberry pi into a power source and the LED should turn on.

Testing Raspberry Pi

- Plug the raspberry pi into a power source and connect it to a monitor.
- You should be greeted by a screen asking you to enter your login information
- If you haven't made any modifications then the default credentials are Username: pi Password: raspberry
- Open the browser and go to any website

  If the pi is connected to internet you should have no problem

Testing the Bluetooth

Assuming the additional libraries were installed you can test the bluetooth by:

- Turn on the raspberry pi, connect it to a monitor and make sure bluetooth is enabled.
- Log in and open the terminal
- Create a new python file, paste this code and save it

```
import bluetooth

target_name = "My Phone"
target_address = None

nearby_devices = bluetooth.discover_devices()
```

```
for bdaddr in nearby_devices:
    if target_name == bluetooth.lookup_name( bdaddr ):
        target_address = bdaddr
        break

if target_address is not None:
    print "found target bluetooth device with address ", target_address
else:
    print "could not find target bluetooth device nearby"
```

- Run it by using the command:

```
sudo filename.py
```

replacing filename with the name of the file you just created.

- It should now look for available bluetooth devices

Testing the motors

- Ensure all connections are made according to the build instructions
- Power up the raspberry pi and login
- Download fwdback.py
- Go to your terminal and go to the directory where you downloaded fwback.py
- Run the program using

```
sudo python fwdback.py
```

- If all connections were made correctly then the wheels should move forward and backward

Testing the sensors

- To test the sensor the steps are similar to that of the motors.
- Power up the raspberry pi and login
- Download sensor2.py or copy the code below
- Open your terminal and go into the directory where you downloaded the program
- Run the program using the command:

```
sudo python sensor2.py
```

- Place an object in front of the sensor and it should return the distance of the object to the screen.

7.2 Product Testing

When unit testing has been complete and all problems resolved, you can now test the car as a whole. Follow these steps to test that all components work together:

- Power up the raspberry pi
- Put in the batteries
- Connect the raspberry pi to a monitor or use remote pc to view the desktop
- Login to the raspberry pi
- Download auto1.py
- Elevate the car so that it isn't touching the ground (You can place it on top of something)
- Go to your terminal and navigate to where you downloaded auto1.py
- Run the program using

```
sudo python auto1.py
```

- The wheels should move forward continuously
- Wave your hand in front of the sensors and the wheels should change direction

You can further test the car by placing it in an open area and watching it go

- Follow the previous steps but use remote pc to connect instead of using a monitor. This is important because you want the car to move freely
- Place the car in the open area
- Run the program again
- The car should move forward continuously
- When the car senses an obstacle it should reverse and change direction(whether it be left or right. This is choses randomly by the program)
- If it senses an obstacle in the direction it turned to then it will turn once again.
- The car will continuously do this until it cannot find a path to continue

7.3 Application Testing

While testing the hardware it would be beneficial to test the application in preparation for the integration of the hardware, software and database. There are multiple tests that can be done to ensure that the application is running as expected. These tests are focused on checking if the application is connected to the database and whether the application is usable.

Testing Signup

The signup activity is used to allow the user to add their information to the database allowing them to login.

an internet connection is required to signup and login

- First you must open the application and ensure that you have an active internet connect on your device
- Tap the text under the Log In button that says sign up
- Enter a first name, user name and a password
- In the final field reenter the password that was entered in the password field

  There are no word limits or text requirements that need to be met as long as all fields have characters in them and the passwords in both fields match.
- If the passwords match and all fields are entered then the user can now tap the sign up button and the information will be sent to the database.
- If any field is left empty and the signup button is tapped an message will be displayed over the empty field and the information will not be sent to the database.
- If signup is successful the user will then be taken back to the login page.

Testing Login

The login activity is the first page the user is greeted with when the application is opened. It allows the user to login and use the application provided that their information exists in the database

- If you do not have login credentials follow the signup mentioned in the procedure above. If you do then enter your user name and password
- Tap the login button
- If the login information matches the information stored in the database then the user will be presented with the Main Menu page
- If the login information does not match the information in the database

Testing Remote Control Activity

The remote control activities are used to control the car remotely from the android device. There are two remote control activities that allow the user to connect to the car using two different connections, namely: Bluetooth and WiFi(internet). The complete functionality of these activities will be done in the integration

- To test both acitivities the user will need to login to the application
- First go the first remote control activity from the main menu

  Because there is no IP address entered the user should see a pop up asking them if they would like to go to settings

  Tap the later button and then tap the connect button.

  The application should then display a message saying that the IP address needs to be entered. If the user taps the close connection button another message should be displayed saying that there is no connection to close
- Next, go back to the main menu and tap the second remote control button under the mapping button.

  The user should again be greeted by a pop up but this time it will say ask the user if they want to turn Bluetooth on in settings

  Tap the later option and try to connect using the connect button.

  The application should display a message saying that Bluetooth needs to be turned on. Additionally if the close connection button is tapped a message should be displayed saying no connection to close.

  Go to settings and turn Bluetooth on or do it through the device's system UI

  Return to the activity and the message should not be displayed

  If the connect button is tapped it should say trying to establish a connection but because there is no device for it to connect to it will continue searching for a sever
- Moving the joystick in any of these activities should display text stating which position the joystick is being moved in.

Testing Mapping Activity

The mapping activity is used to run the autonomous functionalities of the car and display a map of the mapped area on the android device. As with the remote control activity, the complete functionality of this activity will be tested in the integration tests

- Login to the application

- Tap the button that says Mapping in between the two Remote Control buttons
- If Bluetooth is turned off a pop will display asking if you want to go to settings and turn Bluetooth on
- Assuming that Bluetooth was not turned on, tap the start button
- A message should be displayed saying Bluetooth is turned off
- If the close connection button is tapped while Bluetooth is still off a message should be displayed saying no connection to stop.
- Go to settings and turn Bluetooth on or do it through the device's system UI
- Return to the mapping activity and tap the start button, a message should be displayed saying trying to establish a connection but because there is no connection to be made at the moment the device will continue to search

7.4 Integration Testing

Assuming that all other tests have been run successfully the last set of testing required is the integration of the entire system. You should have already tested the connection from the application to the database in the application testing and the functionality of the car in the unit and product testing. This stage is focused on testing whether the system on a whole is integrated and functional

Testing remote control of the car

This will test the connection between the car and application in addition to testing the remote control features that were explored in the remote control activity testing mentioned previously.

- Turn on your raspberry pi and log in
- Open the terminal
- Download bluetoothserv.py, ctest.c and functions.c
- Verify that the bluetooth functionality is working by referring to the Build Instructions and the Bluetooth testing above and ensure that you have followed all the necessary steps
- Go to your terminal and navigate to the directory that you stored bluetoothserv.py in
- Run the command

```
sudo python bluetoothserv.py
```

This program will create a bluetooth server running on an RFCOMM channel, by default you should be on RFCOMM channel 1

If all the necessary Bluetooth steps were followed the terminal should print "Waiting for connections"

- Log in to your application
- Tap the three vertical dots in the top right corner of the screen and from the options menu go to settings
- Turn Bluetooth on
- Go back to the main menu and tap the second remote control button, the one under the mapping button
- Tap the connect button and a message should appear saying trying to connect
- The application should now connect to server running on the raspberry pi, the connection is confirmed by a message in both the application and in the terminal window of the raspberry pi. ON the application a message should appear saying connected. In the terminal window you should see "found a connection" followed by your device's mac address.
- Move the joystick in any direction, if the car moves then the test is successful
- Tap the close connection button to stop the car and terminate the connection

Testing the autonomous mapping functionality

This is a test of the core functionality of the car which is the autonomous operation.

- Turn on your raspberry pi and log in
- Open the terminal
- Go to your terminal and navigate to the directory that you stored bluetoothserv.py in previously
- Run the command

```
sudo python bluetoothserv.py
```

- Log in to your application
- Tap the three vertical dots in the top right corner of the screen and from the options menu go to settings
- Turn Bluetooth on
- Go back to the main menu and tap the mapping button
- Tap the start button and a message should appear saying trying to connect
- The application should now connect to the server
- The car should now be observed moving
- The display the map tap the stop button to stop the car. A message should appear saying the car has stopped and the map should display in the white area above.

Conclusion

This autonomous vehicle is fully autonomous and uses ultrasonic sensors to navigate and record coordinates in an area which can be sent to a database. This database can then be retrieved by a website to display them to specific users when they log in. The mobile application retrieves the coordinates from the database in addition to a drawn map of the area which it will be able to display to the user. The application can also remotely control the car through the usage of an onscreen joystick, through a Bluetooth or WiFi connection. The information presented is be centered on the design of the autonomous vehicle in addition to the mapping algorithm which it implements.

Recommendation

The AutoVehicle is a fully functional project that achieves the main functionality we desired. Upon the replication of this project there are modifications that can be made to improve the project. As such, there are recommendations that we would like to make. Firstly, a power bank can be used to power the motors and H Drive instead of the AA batteries. This will increase the duration of time the motors can be run for and it can also be recharged when not in use. Secondly, there is also much room for improvement in the mapping aspect of this project. The mapping algorithm mentioned in the project is simple and leaves much to be desired. The solution can be refined to either map entire room rather than the perimeter of the room. The Bluetooth code was written in Python for our educational benefit as we had the opportunity to learn more about the language but in the future it would be better and more efficient to code in C instead. Python required that we install an additional module that would not have been necessary in C. It should also be noted that this module was actually was actually written in C. Additionally, the addition of an IMU unit can be used to further improve the project. This IMU unit would contain two accelerometers and a gyroscope which would allow the vehicle to track it's position more efficiently. GPS can also be incorporated to allow the vehicle to be aware of its location in a room. *talk about app and map*

Appendices

References

Gehrig, S. K., & Stein, F. J. (1999). Dead reckoning and cartography using stereo vision for an autonomous car. In *Proceedings 1999 ieee/rsj international conference on intelligent robots and systems. human and environment friendly robots with high intelligence and emotional quotients (cat. no.99CH36289)* (Vol. 3, pp. 1507–1512 vol.3). https://doi.org/10.1109/IROS.1999.811692

Ismail, H., & Balachandran, B. (2015). Algorithm fusion for feature extraction and map construction from sonar data. In *IEEE Sensors Journal* (Vol. 15, pp. 6460–6471). https://doi.org/10.1109/JSEN.2015.2456900

Katzourakis, D. I., Papaefstathiou, I., & Lagoudakis, M. G. (2010). An open-source scaled automobile platform for fault-tolerant electronic stability control. *IEEE Transactions on Instrumentation and Measurement*, *59*(9), 2303–2314. https://doi.org/10.1109/TIM.2009.2034575

Temeltas, H., & Kayak, D. (2008). SLAM for robot navigation. In *IEEE Aerospace and Electronic Systems Magazine* (Vol. 23, pp. 16–19). https://doi.org/10.1109/MAES.2008.4694832