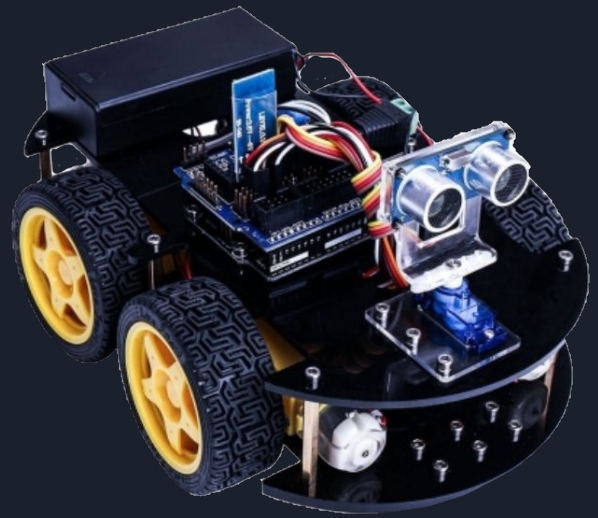
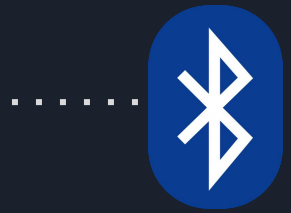
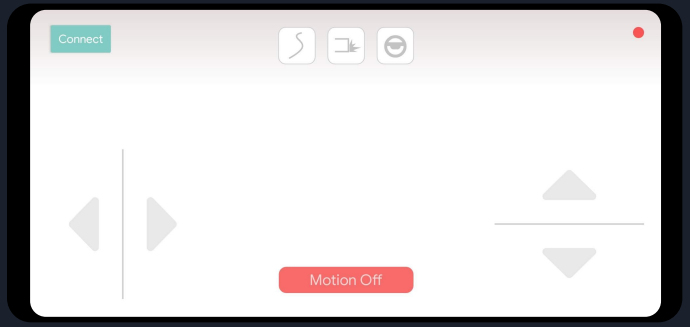
CS M117 Final Project
Bluetooth Car

Fall 2018

Presented by:
Eduard Grigoryan
Wai Yan (Will) Htun
Irfan Sattar
Michael Warren
Ram Yadav
Pyae Phyo (Henry) Zaw






Connect
Motion Off

Introduction

- With many major tech and car companies working on self-driving and autonomous cars, we decided to create a Bluetooth connected Arduino based car that could serve as a simplified model of some of these self-driving cars
- As a result, we have implemented a few different modes/features for the car
  - Manual Control Mode w/ Collision Detection
    - Similar to your car which you drive manually and have computer assistance such as emergency braking, our car can be manually driven with motion control or button controls and while being controlled manually, looks out for possible collisions (In the case of collision, the car will stop, turn around 180 degrees and then wait for manual control from the user)
  - Autonomous Mode w/ Collision Detection
    - This mode mimics the autonomous mode of self-driving cars in that it will keep driving on its own until a collision is detected at which point it will stop, find a path without collision and continue driving on this new path
  - Line Detection Mode
    - In line detection mode, our car will continue to follow a line drawn on the floor similar to how some autonomous cars detect lane markings on roads to stay in lane (In line tracking mode, once the car reaches the end of the line, it will simply spin in place)

Initial Project Proposal

- Manual control that uses the gyroscope/motion control of the phone or button layout to tell the car to go forward, backwards, etc (Manual mode)
    - Collision detection for manual mode where if a collision is detected, the car stops, turns around 180 degrees and sends a message to the app saying a collision was detected; the user can then continue to drive manually
- Autonomous mode that is controlled with an on/off button in the app and the car simply moves forward until it detects a collision at which point it keeps turning to avoid collision and then continues going forward
- Voice control (Extra feature we want to add, but we are uncertain about it).

Changes We Made/Actual Features

- Implemented a majority of the initial proposal; manual control with collision detection (buttons and motion), line tracking mode and autonomous mode using a Bluetooth connection to facilitate communication between our phone and the Arduino car
- Extra features
  - Line tracking
    - The Arduino car will follow a line on the floor and once the line disappears, the car will spin until it detects a line again
    - Line tracking only works in a straight line since one of the sensors on the car broke so making it turn would not work properly; since it broke so close to the due date, we weren't able to order those parts again
- Removed features
  - Voice control (Optional feature)
    - Since we decided to focus on some other features and because of the very limited amount of time because of hardware issues, we weren't able to attempt to implement this feature

Technologies & Parts Used

- Hardware
  - Arduino UNO board, L298N module, Bluetooth module (HC-05), Ultrasonic Sensor (HC- SR04), Motor Shield (Adafruit), Mini servo motor (SG90), DC motor x 4, Wheel x 4, Chassis plate, Holder for HC-SR04, Battery gv x 2, Battery Buckle x 2, Glue Gun, Rocker switch x 4, Tactile Tact Push Button Switch x 4

- Software
  - Android Studio, Arduino IDE
  - We wrote the code for manual control, collision detection, line tracking on the Arduino UNO, and the Android application which we use to communicate with the car
  - Github Link: https://github.com/irfansattar/CS-M117-Bluetooth-Car

Implementation Details

Assembling the Car/Arduino Code

- The car kit came with many different hardware parts which we used to build our final car
- The body frame is assembled with parts from the kit. On the chassis, we installed:
  - Arduino UNO board, bluetooth module, ultrasonic sensors and line tracking sensors
- We coded the logic and movement of the car for different modes on the Arduino UNO
- Implementation of different modes
  - Manual control with collision detection
    - Used the phone's gyroscope sensor to determine rotation about the Y- and Z-axes and if the rotation was strong enough/above a certain threshold, we send a command to the car to go in the appropriate direction
    - If collision is detected, the car stop and turn 180. Manual control is disabled during this period. After fully turning 180, we can perform manual command again
  - Autonomous mode with collision detection
    - Car moves forward, right and left itself.
    - When car detects the hazard, it turns 180 degree and then keep going forward
  - Line tracking
    - Car will follow the straight line and will spin in place whenever it detects the line ending/loses the line.

Implementation Details (Cont.)

Application

- Android application written in Java which connects to the Bluetooth module on the Arduino car
- The connection is established through the Android Bluetooth libraries and makes a connection to the BLE (Bluetooth Low Energy) module on the car
    - For making a connection, we used Android's Bluetooth libraries to have our smartphone's Bluetooth module search for and connect to the BLE module on the car
    - To send and receive data, we overrode the Bluetooth functions that are required for communication with a BLE module to send/receive string data as a byte array
    - Both the phone and car ends, we interpreted the received string accordingly to carry out the appropriate action
- The purpose of the app is to give the user an intuitive way to communicate with the car to perform actions such as maneuvering, switching modes and receiving feedback from the car.

Project Demo

- We have uploaded a demo video on YouTube which can be accessed with the link below.

https://www.youtube.com/watch?v=JgocaLvhb7k&feature=youtu.be

Conclusion

- Overall, the project was completed successfully. We were able to utilize wireless communication using Bluetooth and an Android application to communicate with our car.

- There are some areas for improvement/additional features such as:
  - Line tracking sensor replacement to fully realize the line tracking mode
  - Voice control as an alternative input for manual mode
  - On car LED lights that can be turned on/off and changed color through the app
  - Better pathfinding algorithm for autonomous mode so that the car finds the right path out when cornered or hit a dead end. With this improved algorithm, it should also choose the most efficient path when more than one option is available