**Day 1**

System will control:

* Headlights - LED lights
* Honk - speaker
* Speed - motors (speed control)
* Stop & Go (at stop signs) ?

System will be placed at entrance (to special zone) roads, exit roads, stop signs, etc.

Car will not correspond to actual car in project

How to detect car:

* Lasers
  + If transmitter does not see laser, communicate (how?) information to car
  + Need to find inexpensive lasers
* RFID (FasTrak)
  + Might not be accurate enough for scale model

**Day 2**

Use Bluetooth to communicate with car

Small range (and accurate) RFID can be purchased online - going to use RFID

* Range ~ 2 cm - original testing criteria of 30 cm not going to work (no cost-effective options with close to 30 cm range)
* Active vs Passive tags
  + Active = battery power source to broadcast, Passive = only broadcast when receiving radio frequency energy from nearby RFID scanner
  + Using passive - simpler, has everything needed for project

How to connect RFID and Bluetooth? -> Arduino

* HC-05 and HC-06 to work with Arduino Uno

**Day 3**

Getting basic functions on RFID and BT modules (separately)

Challenges:

* RFID Tag not working
  + Solution: pins were not soldered well - resolder
* Master BT module not going into pairing mode or AT command mode
  + Special button on chip - needs to be pressed while power on

| **Master** | **Slave** |
| --- | --- |
| HC-05 | HC-06 |
| 6 pins | 4 pins |
| Special button for AT command mode + pairing | No special button |
| usbmodem1401 | usbserial-130 |
| Line ending: NL & CR | Line ending: No Line Ending |
| BAUD: 9600 | BAUD: 9600 |

Challenges:

* One slave not accepting AT commands for initialization
  + Slightly different models, got right model

Successfully send message from one module to another

**Day 3**

In model: 2 cars, 2 scanners (the part of the system on the road)

Decided to have 4 bluetooth links - 2 on each car - one connects to each scanner



The acronym that comes first will show if the BT module is on the car or scanner. For example: s1c1 is the scanner in the link that connects the first scanner to the first car.

**Day 4**

Tried to get car to listen to both scanners, was not able to get fully reliable communication

Started doing line follower - did not work

Challenges:

* Tried to listen on both links (c1s1, c1s2), did not get response
  + Added delays, able to get message sometimes
    - Send commands 10 times, 1 will be received
  + Cannot listen to both serial ports at the same time, since hardware port on arduino was divided into 2 software serial ports to accommodate both BT modules
  + Keeps switching between the 2 software serial ports (has to be there at the right time)
  + Delays increase probability, not fully reliable
* Line following not working
  + Work on next day

**Day 5**

Fixing line follower problem

1 = black (light off on sensor)

0 = white (light on on sensor)

100/110 = turn left, 001/011 = turn right, 010 = forward, 000/111 = stop

Line follower getting stuck on curves

* Lines thicker
* “111” set to forward instead of stop
* Fix stuck caster wheel

Line follower fixed after changes in code, could reduce line width as well

**Day 6**

Build 1 car (without the actual LED and speaker, using builtin LED on arduino)

Transfer BT to smaller breadboard, and connect everything to 1 arduino - have to change pin numbers on code

Challenges:

* Not enough space to connect all wires
  + Found a shield to use
* C1s2 BT module not working after transfer
  + Wiring mistake (GND->5V instead of GND)
  + After retesting, everything worked fine (nothing was damaged)

Designed scanner box and mount for RFID tag on CAD software, printed overnight

**Day 7**

Build 1 scanner box

Transfer BT to smaller breadboard, connect RFID scanner and BT to 1 arduino (needed to use shield as well)

Challenges:

* 3d-printed box had some minor issues
  + Reprinted lid so it would fit better, used drill to widen holes (screws were not fitting)
* No 3.3V for the RFID scanner on the daughter shield
  + Found some pins that had 3.3V coming so just used them

BT modules worked fine after transfer, RFID also worked when tested.

**Day 8**

Combine BT, RFID, and line follower code

For BT code: scanner sends string of characters over BT to car for commands

Need to find a way to find the first correct string

* Put unique letter in front of each string, if see that letter then it is valid string

|--Byte--|

| 0xFF | - | [light] | - | [sound] | - | [speed] |
| --- | --- | --- | --- | --- | --- | --- |

Not working: Car is not acting based on commands

Debug

* Car is not able to display valid commands in serial port, but scanner is sending commands correctly
* Display all commands (valid & invalid) -> unable to receive strings and display those
* Disabled line follower in case it was disrupting -> was then able to display strings

**Day 9**

Kept working on car commands

Debug (continued)

* Change some logic in code, and letter at the beginning of strings -> was able to receive valid commands, however, car kept stopping and lights didn’t work.
* Figured out characters for changing the light and changing the speed were mixed up, so it wasnt working correctly - after fixing, was able to successfully follow commands

**Day 10**

Spray painted the board for the track (currently it was unpainted and the line was black tape)

Challenges:

* White spray paint was not working
  + used other white paint and paint roller
* The tape for the track kept coming off, so the black line was not very precise
  + Did touch ups with sharpie afterwards

**Day 11**

Build second car, test with 2 cars and 1 scanner

After building second car, had to alter scanner code to be able to detect 2 different cars

Worked first try - while debugging the command issue, also made scanner code able to accommodate 2 cars, so no problems

**Day 12**

Built second scanner box (could not test, unexpected issues)

Had to configure c2s2 and s2c2 first - unexpected issues took a long time to resolve

* Could not establish BT connection
* After redoing process and changing many other things, realized the arduino was bad (used a new spare one as to not touch the completed cars)
  + Decided to replace c2s1 on car 2 with c2s2 to configure

After building scanner box, had to edit code on both cars

Only had to copy and paste a portion of the code, as well as slightly edit it

**Day 13**

Tested second scanner

s2c2 link was not working - car 2 would just skip over the commands sent to it

* BT connection was not working
* Checked code and found an issue, however there was still no connection between the 2.
  + There was also a loose wire in the scanner box, after fixing it all the connections worked

Cars have different speeds, keep bumping into each other

* Solution: create a stop system at each scanner

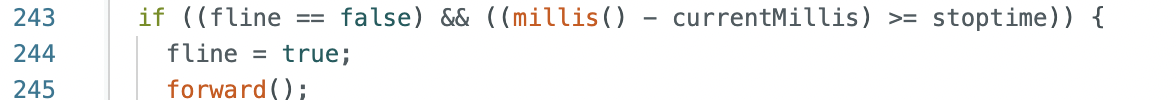
**Day 14**

Stop idea:

* Give each car N number of seconds to make it from one scanner to the next (sent over by the character in the commands labeled “stop”)
* Car will start timing when it receives command from one scanner, and record timestamp when it reaches second scanner
* Subtract that time from N seconds, and wait at the second scanner from that new time.
* This way, cars will not bump into each other - can also be used as another command (ex: stop signs, crosswalks, signals, etc.]

Tried using wait function for stopping

* Line follower started stopping randomly, and BT commands became delayed
  + Wait function disrupts other parts of code from happening
  + Used Stop function instead of wait function, and used conditions to determine when to start moving again



millis() - used for getting timestamps

Since counting with car was getting too complicated, tried to use scanner instead

* Scanner count to 10, and every 10 seconds send out a 1: car does not move until it receives a 1

**Day 15**  
Tried scanner idea, altered scanned code

After testing, cars were not stopping

Debug

* Temporarily changed code on scanner to display everything being sent
  + Monitored scanner, it was sending a 1 every 10 seconds
* Checked serial port on car
  + There was not a consistent flow of 1s - much longer than 10 second wait, and was not constant

Could not figure out root of the problem, easier to go back to old method

**Day 16**

After many more changes to the logic, car was finally stopping consistently at every scanner - while using a set value as the stop time (was not getting it from the command string sent by scanner)

Using the character from the scanner string did not cause any problems, worked the same way as before.

**Day 17**

After getting stop procedure working, did a few test runs with the built in LED to see if the system would work all together - no problems (woohoo!)

Instead of using the built in LED, got an actual one and tried using that

However, only pins left were the analog pins - had to check if they worked with those

* They worked fine as well with analog pins, so was able to get an actual LED on both cars instead of just the builtin LED

**Day 18**

Tried to get speaker working as well

Speaker was also able to work with analog pins

Challenges

* Functions were not working
  + Tutorial was using a different kind of speaker, found another tutorial
* Speaker was not working at all
  + Learned about PWM pins - had to rewire so that everything could run properly
* Speaker was disrupted (turning on and off very fast)
  + Changed placement in the code - was able to make it work
* Car 1 stopping randomly, going very slow
  + Batteries ran out, put new batteries in

PWM pins - used to control voltage (create different speeds)

* Sets voltage to high and low very fast, in order to average a certain speed (normally, voltage is either high or low - no in betweens)
* Helps with things like speed in DC motors

Made a few final small changes (added top covering to cars, etc.)

DEMO DONE!!

**Day 19**

Testing - passed, failed

Criteria 1

Trials: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

100% pass

Criteria 2

Trials: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

100% pass

Criteria 3

Trials: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

100% pass