Autonomous Automobile

**Problem Definition:**

In the real world, there are numerous lives lost due to weary or unskilled drivers. This problem can be completely eliminated or at least reduced if driving were to be autonomous. There are many companies such as google and Volvo working on such technologies. However, the project using the LEGO NXT kit will clearly supersede their efforts!

Obviously, every possible situation in the real world cannot be simulated due to technological and resource limitations. As a result, the following aspects of an autonomous vehicle have been chosen to be executing a set number of tasks that shall attempt to imitate on road and traffic scenarios.

**Criteria, Constraints and Task List**

In the problem, defined above, it was established that the robot was to automate the task of driving. As per the rubric, the constraints that must be fulfilled in order for the mentioned problem to be solved are navigation, lane guidance, traffic lights, and collision detection.

The navigation system will be created to reach predetermined destinations by hard coding using motor encoders. A lane guidance system, using light sensors, will enable the robot to follow lane markings making sure that it will stay in the lane. The traffic light system will be based on color stimuli (possible paper cards) to make sure the car stops or continues at a traffic junction. A collision detection system will be based on the robot immediately stopping after a collision takes place and responding to the crash by “calling” emergency services. A further added safety factor is the ultrasonic sensor based obstacle detection that will ensure no unexpected obstacles (i.e. jaywalkers) will lead to accidents.

The criteria set forth for this project will further enhance the autonomous properties of the vehicle. Collision evasion is accomplished by swerving around the obstacle and return to its predetermined lane. An automated parallel parking system can be created to park the robot between two stationary objects. Similarly, another parking system can be used to help reverse the robot into a garage at a destination. Finally, the gas cost will be displayed on the robot’s screen at a destination using motor encoders and presumed efficiency values.

**Task List:**

* Navigation from set point to desired location
* Guides itself in a predetermined lane
* Responds to traffic light operations and pedestrians, if necessary.
* After reaching a set destinations, displays an approximate fuel cost
* If involved in a collision, halts, warns user and ends program

**Functions being implemented:**

void traffic();

* Using the color sensor to stops the robot until the “traffic light” turns green and there is no pedestrian in the way.

float fuelPrice ();

* Using encoder counts, return the distance covered multiplied to the price of fuel 0.5$/cm

void printPrice(float fuelPrice, float package);

* Calling fuelPrice and the cost of a package to display cost to customer

void tippy();

* Using the motor encoders, tip tray over to drop package at destination

void steer(char command);

* Takes in either a ‘s’,‘l’ or ‘r’ and completes the corresponding turn.
* This function uses the motor encoder designated for steering.

void linefollowing (bool house);

* Using the color sensor this function is to guide our “car” in a lance that has been outlined.
* The function will call the above steering function to execute turns at junctions
* When the road reaches the lane with the house the light sensor will switch on and search for the house

**Program testing procedure**

Function testing:

traffic () –

* Have yellow and blue paper arranged in series, the car should stop once at the blue paper.
* Swap the blue paper for green, the car should proceed.

fuelPrice() –

* Have the car move forward till a certain predetermined distance, calculate price by hand and compare to the result round by the bot.

printPrice() –

* Have the car print for the package price, fuel price and the total of both

Steering –

* Call the function in lane guidance with ‘s’,‘l’ and ‘r’, and check for the following to occur:
  + The front wheels steer right/left to full lock without slipping of the gear.
  + If ‘s’ the car should carry on forward.
  + Wait at that position for a pre-determined amount of time.
  + Return to the “original” straight position.

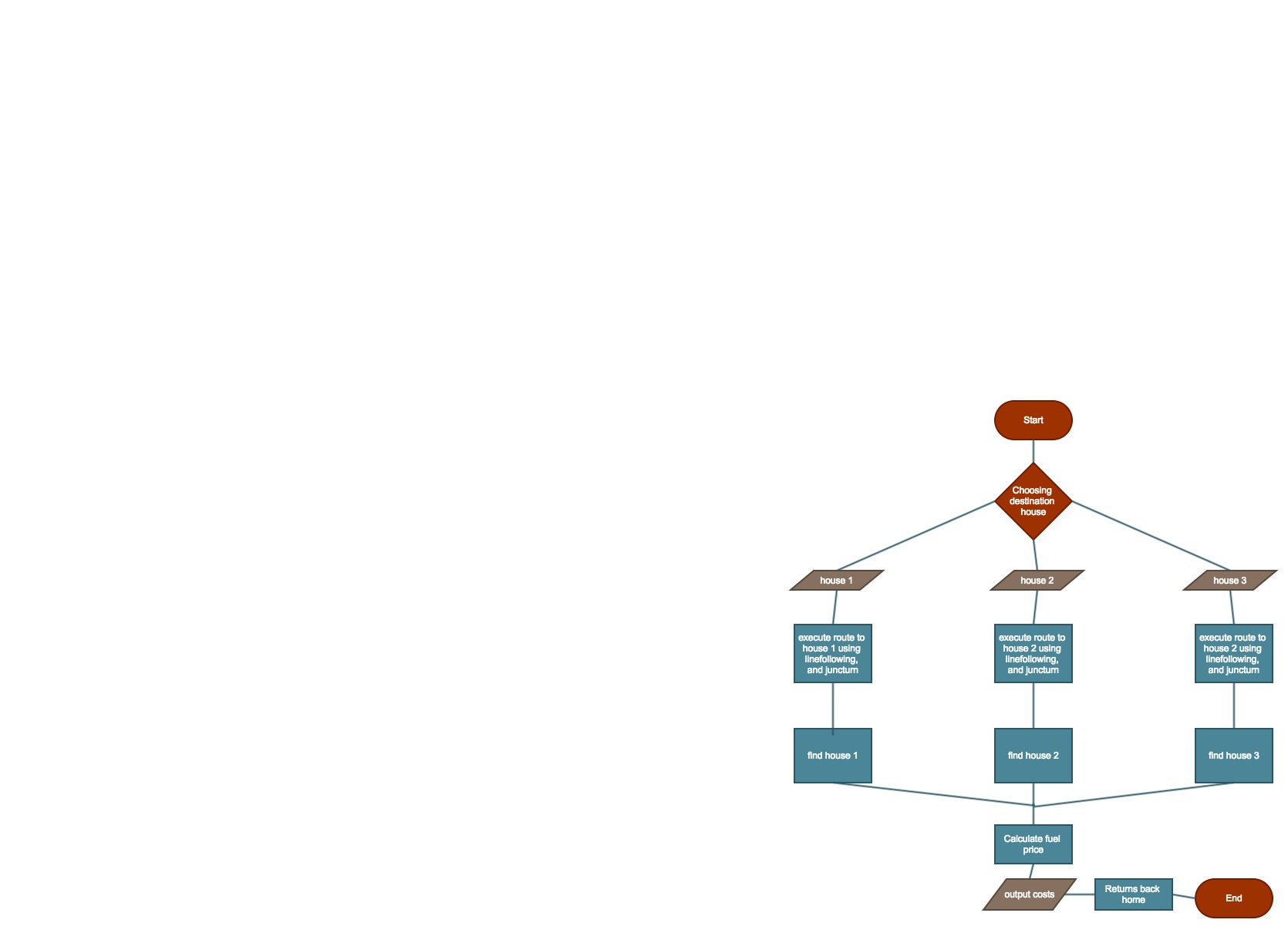
(This will be performed with the robot on the ground and no drive)

Line following (bool house)–

* Make a sample lane with black tape and yellow and red borders for a right and left curve to test if the robot can detect the yellow borders of the lane.
* If the inside or outside of the lane is detected, it should turn by calling the steering function until it detects the yellow border again -whereby it will straighten itself to follow it.
* Also, the detection of the house can be tested by placing an object in front of the car which will stop the car and disembark the package

Testing the whole Program

* The robot will be driving around on white chart paper with roads marked out with a black strip indicating the inside lane of the road and the yellow borders of the lane indicating the line that the robot needs to follow. The steer and linefollowing function will be tested in this phase.
* There will be a junction at which the robot will have to come to a complete stop if the strip of paper it detects is blue. If the strip of paper is green, the robot will proceed as before and turn in a specified direction unless there is a ‘pedestrian’ crossing in front of the robot and the signal is green. The traffic light and steering function will be tested in this phase.
* The robot should display a message showing the cost of the package and fuelPrice and use tippy to tip over the package.
* If either of the above functions do not work as planned, debugging will be used to resolve the issues until the robot will perform the above functions.



**Conclusion**

Though the course of the project there were several changes that were made which occurred due to how the sensors behaved. The most impactful would be that when the color sensor sensed yellow it would often also sense it as red, which was used in the signal for the traffic thus stopping the car. Before the problem arose, blue color was used to indicate the location but now it had to be employed as the red signal. With all the colors, available by the color sensor being used up, a light sensor was added on to detect the house. Also, the crash function was removed to retain simplicity of code, as it did not behave as expected. Furthermore, functions were added to the program to initialize, to end and to pick houses.