# Introduction

The Freenove Smart Car was chosen as a raspberry pi platform for introducing interns to embedded software and artificial intelligence concepts. Taking advantage of the sensors and peripherals provided by the platform, the following projects were determine to provide an adequate introduction within the time afford the projects. Each project is tailored for a one week duration allowing interns time to both meet project objectives along with completing the other tour, panels, and briefings there are assigned.

# Projects

The Freenove Smart Car provides sample python client and server code for implementing line tracking, object avoidance, face tracking, utilizing the LEDs, and other features.

(*Note: The code provided by Freenove contains updates supporting a newer camera. The picamera2 import and respected code needs to be reverted to utilize the legacy picamera libraries.*)

[**P001**] The requirement for each of the following projects is to implement the functionality in the raspberry pi. Outside of any code for starting, stopping, resetting, etc., the functions performing the below requirements should reside in the car.

* *Note: Streaming video to the client application to include object detection bounding boxes would be preferred as it would allow users to see the system work, but should not be necessary for the car to perform the requirements.*

[**P002**] The application shall use Python

[**P003**] The application shall use YOLO for its AI object detection engine

- *Note: Recommend Roboflow for picture annotation.*

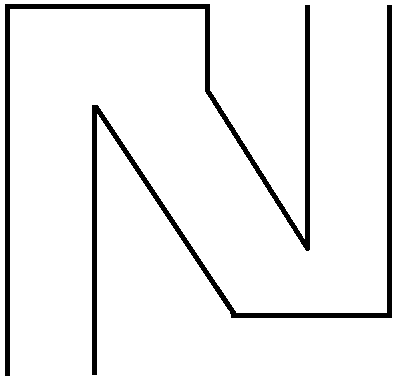
[**P004**] Deliverables shall include source code git repository, install instructions (pip, apt, etc.), scripts used to train AI model

## Project 1

*Project 1 focuses on understanding the motion of the vehicle and its sensors by having the car traverse a track.*

[**P1-001**] Navigate a course (*sample course below*) by utilizing the infrared sensor to perform boundary detection and stay within the course (*Note: boundary detection and not line tracking*)

[**P1-002**] Car shall stop when the ultrasonic sensor detects an object which can be placed at the end of the track



*Figure p1: Sample Navigation Course*

## Project 2

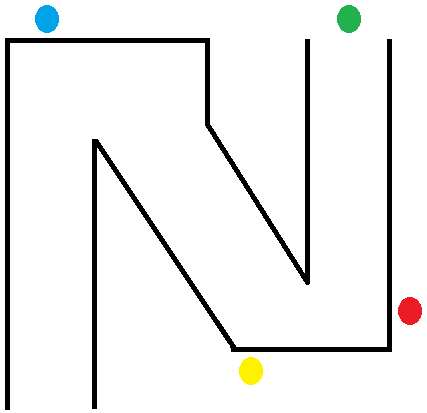
*Project 2 expands upon project 1 by introducing the camera and basic object detection.*

[**P2-001**] Implement requirements P1-001

[**P2-002**] When the software detects a ball within one foot, the car shall perform the following activities

* Determine color of ball and change vehicle LEDs to match color of ball
* Pause the car for five seconds before continuing down the track

[**P2-003**] Once a ball color is detected, the application shall ignore that color ball for the remainder of the program.



*Figure p2: Sample Navigation Course w/ Balls*

## Project 3

*Project 3 focuses more on object detection. The project goal is to detect balls of differing colors and knock them outside of the boundary. The application should find and remove balls in a user provided sequence.*

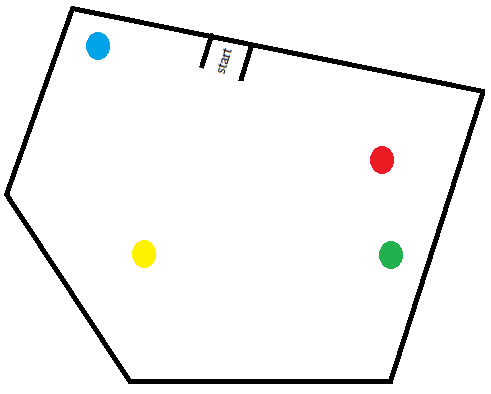
[**P3-001**] User shall input a four ball color sequence used to determine the order in which the application should detect and knock out balls.

[**P3-002**] The car shall remain within the boundary course (*sample boundary below*)

* *Note: Car must remain within the boundaries of the course and cannot utilize the line tracking feature for finding the balls*

[**P3-003**] The application shall detect (i.e. find) balls, one at a time, in the order provided. The car shall perform the follow activities once a ball is detected.

* [**P3-003-01**] Change vehicle LEDs to match color of detected ball
* [**P3-003-02**] Drive car into ball to knock ball outside the boundary. If ball remains inside boundary, application should continue attempting to knock ball outside of boundary until a boundary wall is detected
* [**P3-003-03**] Flash LEDs to indicate condition has been met prior to transitioning to next ball in sequence. LEDs should reset to off once after three seconds of indicating P3-003-2 requirement has been met.



*Figure p3: Sample closed course*

## Project 4

*Project 4 expands upon the functionality implemented in the previous projects. Instead of balls, a three class custom AI model should be created. The car will search for each object in the model and indicate the object has been detected.*

[**P4-001**] Create a three class YOLO object model. (*Sample size should be around 150-200 images per object taken in various backgrounds and angles. Sample size will be determine by the complexity of the chosen object.*)

[**P4-002**] Car shall stay within the provided boundary course (*sample below*)

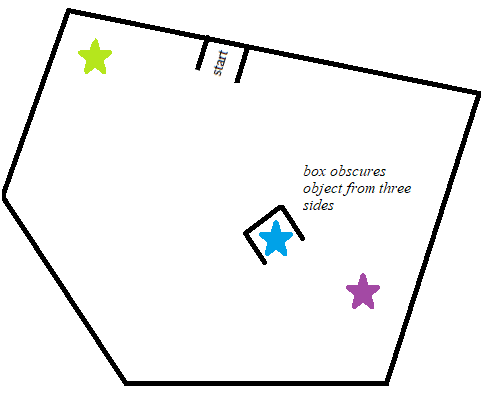
* *Note: Car must remain within the boundaries of the course and cannot utilize the line tracking feature for finding the balls*

[**P4-003**] Car shall avoid objects within the provided boundary course

[**P4-004**] Car shall detect the objects found in its object model. Upon detecting an object the following should occur

* [**P4-004-01**] Flash LEDs in a pattern indicating object found
* [**P4-004-02**] Drive towards detected object and stop between one to two feet away from object
* [**P4-004-03**] Flash LEDs indicating movement towards object has been completed
* [**P4-004-04**] Pause program from five seconds before continuing onto next object

[**P4-005**] Application completes once all objects are detected and steps in P4-004 are completed



*Figure p4: Sample Object Course*

*Note: Optional additional requirement would be to utilize TensorFlow Lite instead of YOLO for exposure to different AI detection models.*