# Autonomous Vehicles Project

# ECE-CSE 434

# Fall Semester 2024

# Team Members

Aldo Madrid

Michael Nguyen

Renu Kulkarni



# Project Initial Plan: Follow a Lane Instead of a Line with the TurtleBot

## Problem Summary

One of the core challenges in autonomous navigation is accurately detecting and following lanes on roads or paths rather than simple lines. While line-following algorithms can guide a vehicle along a single line, real world scenarios usually require navigating within a full lane, which could include dashed or solid boundaries and complex curvature. The TurtleBot - a widely used robotic platform - can be adapted to follow a lane using a more sophisticated perception approach that mimics autonomous vehicle behavior.

## Importance of the Problem

Lane following is a fundamental capability for autonomous vehicles as it ensures the vehicle stays within the driving path, avoiding adjacent lanes, curbs, or off-road areas. In real world applications, effective lane following helps improve safety and reliability during autonomous navigation. Implementing this capability on a TurtleBot provides a practical simulation for testing algorithms that can be scaled for use in larger autonomous systems.

## Approach

Note: These are the approaches the team is considering and might be changed based on future work or development.

### Control Module

PID Controller: Design a Proportional-Integral-Derivative (PID) controller to adjust the Turtlebot’s steering angle to follow the detected lane center. The PID gains (Kp, Ki, Kd) will be tuned to minimize lane deviation and ensure smooth navigation.

Velocity Control: Adjust the speed of the Turtlebot based on the curvature of the lane to enhance stability and safety.

### Simulation and Testing

Gazebo Environment: Create or use an existing simulation in Gazebo that features a road with lane markings to test the system.

Real-Time Visualization: Use RViz to display the lane detection results and path-following behavior, showing both the detected lanes and the Turtlebot’s response.

### Perception Module

Camera-Based Lane Detection: Utilize a camera module to capture video feeds and apply computer vision techniques to detect lane markers. Techniques may include edge detection (e.g., Canny edge detector) and Hough Transform to identify lane boundaries.

Color Filtering and Region of Interest (ROI): Apply color filtering to isolate lane colors and set an ROI to reduce noise and processing overhead.

### Lane Tracking Algorithm

Lane Detection Model: Use a pre-trained deep learning model (e.g., a lightweight CNN trained on lane datasets) to enhance the robustness of lane detection under various lighting and weather conditions.

Path Planning: Implement a line fitting algorithm (e.g., polynomial regression) to create a path that approximates the center of the lane. This path will serve as the target trajectory for the Turtlebot.

### Challenges and Solutions

Lighting Variations: Implement adaptive thresholding or augment the dataset for training to include various lighting conditions.

Lane Occlusions: Use predictive algorithms that infer the lane path when parts of it are temporarily obscured.

## Description of ROS package.

In order for a simulated TurtleBot to follow a lane next nodes, that are described in the paragraphs below, are proposed to manage the robot’s position, perception of the environment and control.

### Simulation Node.

It will simulate the TurtleBot in an environment with defined lanes. It will publish sensor data like camera images and odometry.

### Camera Node.

This node will provide feed from the simulated TurtleBot camera for capturing the environment's visual details needed for lane detection.

### Lane Detection Node.

It will provide the feed to identify lane boundaries and calculate the centerline through image processing techniques like edge or color detection.

### Localization Node

Uses odometry data from Gazebo to determine the TurtleBot’s position and orientation relative to the lane.

### Path Planning Node

Creates a smooth trajectory for the TurtleBot to follow, using the detected lane centerline and the robot's current pose. These nodes work together to ensure the TurtleBot generates and follows an accurate path while staying within the lane.

### Control Node.

A PID controller is suggested to adjust the TurtleBot's twist, position and speed, ensuring it stays aligned with the planned path.