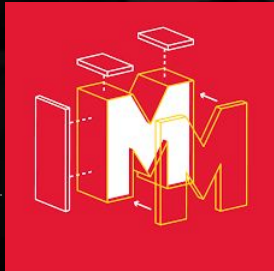# *ENPM- 700:*

Ackermann Steering Controller for Autonomous Vehicle

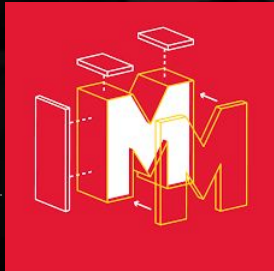Dhairya Shah | UID: 120235146
Harsh Senjaliya | UID: 120215575

Overview

Overview

- Design and development of a control module.

- Focuses on computing steering angles with constraints (max 45°).

- Ensures velocity control and trajectory accuracy.
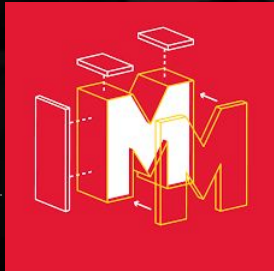
Core Assumptions

Core Assumptions

- **Input from the navigation module.**

- **Focus solely on steering and wheel velocity.**

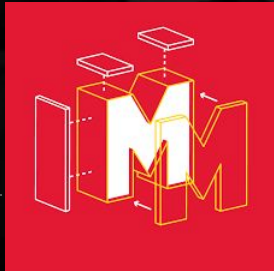- **Utilizes sensor feedback for error calculation.**

Development Process

Development Process

- **Agile methodology with iterative development.**

- **Test-driven development (TDD).**

- **Continuous integration with GitHub Actions.**

Main Functionalities

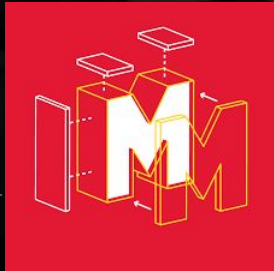# Main Functionalities

- Steering control.

- Velocity control.

- Error minimization using PID controller.

Risk and Mitigation

Risk and Mitigation
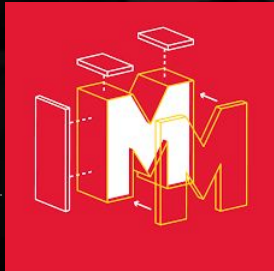
- **Integration challenges (mitigated by thorough testing).**

- **Team coordination (mitigated by pair programming).**

- **Physical limitations of vehicle performance.**
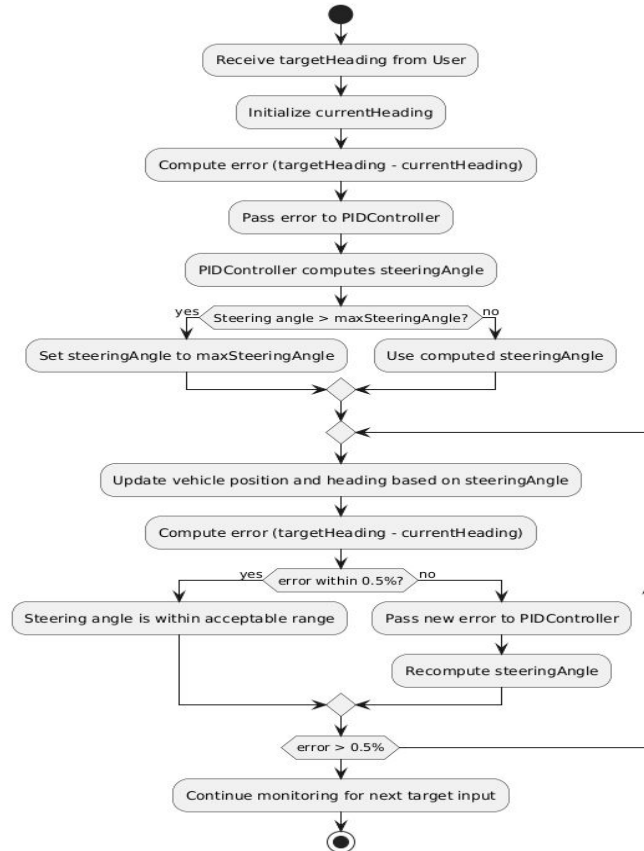
Final Deliverables

Final Deliverables

- **Source code for Ackermann Steering Controller.**

- **UML diagrams.**

- **Unit and system tests.**

- **Documentation**.

Activity Diagram

Activity Diagram

Class Diagram

Class Diagram

Vehicle

- ◻ AckermannController controller
- ◻ double heading

- ● Vehicle(AckermannController controller)
- ● void update(double targetHeading)

AckermannController

- ◻ PIDController headingController
- ◻ double maxSteeringAngle

- ● AckermannController(double Kp_h, double Ki_h, double Kd_h, double maxSteeringAngle)
- ● double computeSteering(double targetHeading, double currentHeading)
- ● bool isSteeringAngleWithinErrorMargin(double actualAngle, double targetAngle, double margin)

PIDController

- ◻ double Kp
- ◻ double Ki
- ◻ double Kd
- ◻ double error
- ◻ double integral
- ◻ double derivative
- ◻ double prev_error

- ● PIDController(double Kp, double Ki, double Kd)
- ● double compute(double target, double actual)
- ● void reset()