**NAME:** FARHAN AHMED, ALI AHMAD BUTT

**ROLL NO.:** 22I-1200, 22I-2385

**SECTION:** E

**DEPARTMENT:** BS CS

**COURSE:** AI

**INSTRUCTOR:** MAM MARYAM SHEHBAZ

# FINAL REPORT

## Introduction

In this project, the goal was to develop an autonomous controller for the TORCS racing simulator that can drive fast, remain centered on the track, and smoothly negotiate turns. We explored both data-driven and rule-based approaches, finally converging on a proportional-derivative (PD) controller tuned via iterative simulation. This report details the data processing, model training experiments, controller design, and tuning methodology.

## Data Collection and Preprocessing

Initial experiments leveraged human-driven telemetry data collected over multiple laps on various tracks. Raw CSV logs contained sensor readings (track position, speeds, angles, range-finder distances) and corresponding control signals (accelerate, brake, left/right steering). We consolidated multiple files into a single dataset by unzipping and concatenating CSVs, then applied cleaning steps: removing initialization frames (gear == 0), discarding stationary samples (speedX $\leq 5$ km/h), dropping duplicates, and balancing steering classes (downsampling straight segments to match turns). The cleaned dataset ensured the model saw an equal representation of left, right, and straight maneuvers, mitigating bias.

## ML Model Training

We trained a Random Forest–based multi-output regressor for continuous control (acceleration, brake, steer) and a classifier for gear selection. Features included 14 telemetry variables plus 19 range-finder readings. Training employed scikit-learn's MultiOutputRegressor with 50 trees and parallel jobs for speed. Despite achieving reasonable performance metrics in cross-validation, the ML policy exhibited oscillatory behavior on oval tracks and occasional stalls, indicating that the decision boundary lacked temporal continuity and anticipatory context.

## Rule-Based Controller Design

To achieve robust, low-latency control, we implemented a PD-based rule controller that directly maps sensor inputs to control commands each timestep. Key components:

- Proportional Steering:
  Steering command = $-K_p \times$ trackPos, clamped to the vehicle's steer lock (±45°).
- Derivative Damping:
  A derivative term (K_d × ΔtrackPos) reduces overshoot and oscillations by penalizing rapid error changes.
- Deadband:
  Steering remains zero within a small center zone ($|\text{trackPos}| < 0.02$) to avoid micro-corrections.

- Adaptive Throttle:
  Base throttle (90%) is scaled down by up to 40% proportional to steering magnitude, ensuring speed reduction in sharp turns.
- Fixed Gear:
  First gear enforced to guarantee forward motion and eliminate shifting anomalies.

## Tuning and Results

Controller gains ($K_p = 0.6$, K_d = 0.1, deadband = 0.02) were tuned iteratively: adjusting $K_p$ improved smoothness, while K_d effectively damped residual oscillations. The adaptive throttle scheduler maintained high speed on straights (0.9 throttle) and reduced to a 0.3 minimum in turns. On the oval track, this PD controller exhibited stable lap times without stalling or zig-zag motion, outperforming the ML policy in both consistency and computational efficiency (sub-millisecond control-loop latency).

## Conclusion

By combining classical control theory with iterative data analysis, we developed a responsive and reliable TORCS controller. Although ML methods provided useful insights into control patterns, the final PD-based approach delivered superior real-time performance and predictability. Future work could integrate ML-augmented tuning or model predictive control (MPC) to further enhance lap time optimization.