Title

TrackMaster: Vision-Based Deep Reinforcement Learning for Real-Time Autonomous Control in TrackMania

Project Partners

1. Harshavardhan Patil - hcpatil@umd.edu
2. Mohit Saluru - mohit95@umd.edu

Goal

To develop an agent that can reliably complete a minimum of three complex randomly chosen tracks using only visual input, demonstrating superior robustness against environmental changes and good generalization to previously unseen tracks

Simulation Platform

We plan to use rtgym based tmrl which has the standardized Gymnasium API. tmrl has built-in wrappers that offer different modalities like raw screenshots and LIDAR, which should streamline the data pipeline. It's specifically built for the newer TrackMania (2020) which might require more computation. An alternative could be TMInterface which could be used for the much older TrackMania Nations Forever.

ML/RL Method

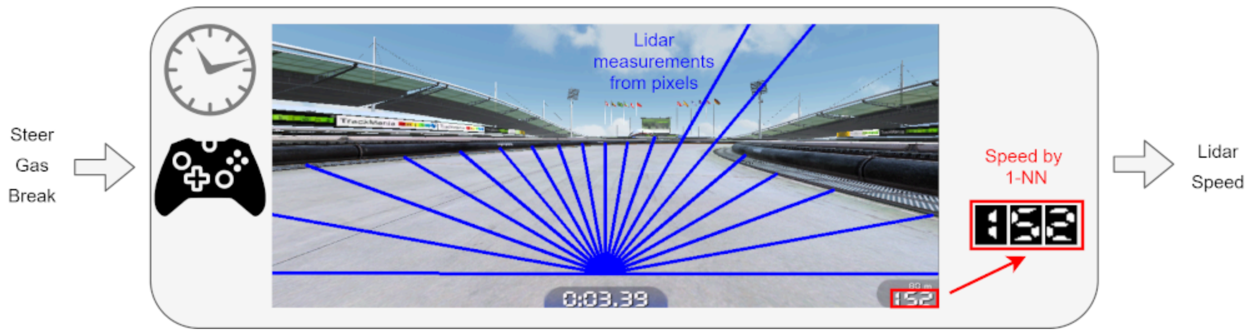
We plan to use Proximal Policy Optimization (PPO and configure it to handle discrete action spaces

Programming Language

Mainly Python, C++ if needed

Project Description

Robot Simulation Platform, Control Methods, and Libraries:
The simulation platform will utilize the TMRL framework and the Real-Time Gym (rtgym) environment. This platform specifically provides TrackMania Gymnasium environment for TrackMania 2020. We will output discrete actions corresponding to keyboard/discretized analog inputs for gas, brake, steering. The observation pipeline will use tmrl raw screenshots captured in real-time. The base for this library uses PyTorch and Stable-Baseline3



Method of Robot Control:
The control method will be Direct Policy Mapping. The agent will map the current visual state directly to the optimal discrete action. The rtgym framework will be used to maintain a reliable deadbeat control loop and mitigate inherent delays

How the ML/RL Technique Will Be Applied:
We will employ Proximal Policy Optimization (PPO). A Convolutional Neural Network (CNN) or Vision Transformer backbone (probably pre-trained and fine-tuned) will process the raw screenshot input to extract key features like track curvature, car position, other cars, etc. These visual features, combined with vehicle telemetry (speed, angle) forms the observation space for our agent. The PPO policy network will then map this observation to a probability distribution over the discrete control outputs. The reward function will be designed to maximize horizontal speed, minimize distance from ideal racing lines and avoid obstacles.

Justification:
We use PPO because it provides a highly robust and proven baseline for policy gradient methods in racing simulations. While off-policy methods like Implicit Quantile Network (IQN) or Soft Actor-Critic (SAC) have been shown to perform better for TrackMania runs, PPO is more stable and easy to tune, which is critical for dealing with the complexity of real-time CV processing. Depending on the time-frame, we might also experiment with SAC.

Fallback Plan

We have a two-fold fallback plan

1. Observation Simplification: If the vision based models fail to train or have excessive latency, we abandon using raw screenshots in favor of LIDAR or direct telemetry.
2. Proven Algorithm: We have found working implementations of SAC/DQN based algorithms. This should guarantee a working bot.

References

Improving Trackmania Reinforcement Learning Performance: A Comparison of Sophy and Trackmania AI

A Driving Model in the Realistic 3D Game Trackmania Using Deep Reinforcement Learning[v1] | Preprints.org

AndrejGobeX/TrackMania_AI: Racing game AI

RAL21_Fuchs.pdf

trackmania-rl/tmrl: Reinforcement Learning for real-time applications - host of the TrackMania Roborace League

Linesight-RL/linesight: AI Plays Trackmania with Reinforcement Learning