

Median Agent Technical Datasheet - Autonomous driving in Super Tux Kart using Reinforcement Learning

Authored by: Wilson Cedric GENEVIEVE
Safa GUNES
Mahmoud DARWISH
Badr BENHAMMOU

Bachelor in Computer Science, Sorbonne University, Paris, France
January 2025 - May 2025

Supervised by: Olivier SIGAUD
Computer Science Professor and Machine Learning Researcher
ISIR Robotics Laboratory, Sorbonne University

Abstract. Our MedianAgent navigates SuperTuxKart tracks using curvature control, adaptive acceleration, and path-following methods to balance stability and speed in complex racing environments.

1 Introduction

The primary objective of our MedianAgent is to stay in the middle of the track while maintaining optimal speed through various track configurations. Unlike conventional racing agents, the MedianAgent relies on the *n-th node* path-following technique to determine its position on the track. This approach involves selecting a future node ahead of the kart's current position as a target for movement.

Initially, we attempted a simpler strategy of selecting the immediate next node as our target. However, we discovered that using an *n-th* node (a node farther ahead) improved steering precision and reduced reaction delays. The following diagram illustrates this improved approach:

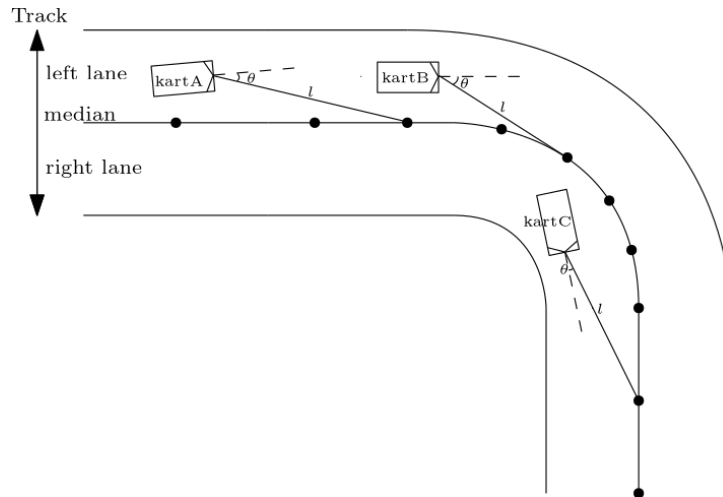


Figure 1: n-th node path-following approach

This represents the most basic implementation of a custom agent, with significant room for improvement, particularly in smoother turns and more efficient trajectory planning.

The challenge with this approach lies in accurately adjusting acceleration, braking, and steering without visual cues or obstacle detection. Our implementation leverages the track's path nodes to create a stable racing strategy.

2 TrackUtils and Path Analysis

The `TrackUtils` script plays a key role in visualizing and analyzing race data. It introduces two key functions:

Algorithm 1 ComputeCurvature Algorithm

```

1: Input: List of track nodes
2: Output: Curvature value
3: for each pair of consecutive nodes do
4:   Compute  $dx = node[i + 1].x - node[i].x$ 
5:   Compute  $dy = node[i + 1].y - node[i].y$ 
6:   Compute  $angle = arctan2(dy, dx)$ 
7:   Append  $angle$  to direction_changes
8: end for
9: if length of direction_changes > 1 then
10:    $curvature = mean(diff(direction\_changes)) \times 10$ 
11: else
12:    $curvature = 0$ 
13: end if
14: return  $curvature$ 

```

Algorithm 2 ComputeSlope Algorithm

```

1: Input: List of track nodes
2: Output: Slope value
3: Extract node1 and node2 from node list
4: Compute  $dz = node2.z - node1.z$ 
5: Compute  $dx = node2.x - node1.x$ 
6: Compute  $dy = node2.y - node1.y$ 
7: Compute  $distance = \sqrt{dx^2 + dy^2}$ 
8: if  $distance == 0$  then
9:    $slope = 0$ 
10: else
11:    $slope = dz / distance$ 
12: end if
13: return  $slope$ 

```

3 The MedianAgent Algorithm

Our `MedianAgent` leverages the `paths_end` values from the observation dictionary provided by the SuperTuxKart environment. The agent selects a node ahead of its current position using the following logic:

Algorithm 3 CalculateAction Algorithm

- 1: **Input:** Observation data, Lookahead value
 - 2: **Output:** Action tuple (steering, acceleration, drift, nitro)
 - 3: Compute curvature using $\text{compute_curvature}(\text{paths_end}[\text{lookahead} - 1])$
 - 4: Compute slope using $\text{compute_slope}(\text{paths_end}[: \text{lookahead}])$
 - 5: Determine direction to target using $\text{direction_to_target} = \text{path_end} - \text{kart_front}$
 - 6: Compute steering: $\text{steering} = 0.2 \times \text{direction_to_target}[0]$
 - 7: Adjust acceleration: $\text{acceleration} = \max(0.1, 1 - |\text{curvature}| + \max(0, \text{slope}))$
 - 8: Activate drift if $|\text{curvature}| > 40$
 - 9: Activate nitro if $|\text{curvature}| < 0.02$
 - 10: **return** ($\text{steering}, \text{acceleration}, \text{drift}, \text{nitro}$)
-

Note: The constants used in lines 6, 8, and 9 were chosen arbitrarily through trial and error. They were adjusted iteratively to achieve stable and efficient driving behaviour in most conditions.

4 Experimental Results and Discussion

Our MedianAgent demonstrated significant improvements in completing race circuits with enhanced stability. By adjusting acceleration in uphill segments and enabling drift in sharp turns, the agent was able to outperform previous agents in maintaining control. However, steering performance in extremely narrow curves remains a limitation.

5 Conclusion and Future Work

The MedianAgent represents a foundational implementation for autonomous driving agents in SuperTuxKart. Future enhancements should primarily consider adding adaptive path selection strategies and improved obstacle avoidance for enhanced racing capabilities.