ItemsAgent 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. The ItemsAgent is designed to avoid all items while navigating SuperTuxKart tracks. While its core logic builds upon the MedianAgent, the ItemsAgent incorporates peripheral vision and customized steering adjustments for item avoidance. As of now, the agent is not yet functional, and no extensive tests have been conducted.

1 Introduction

The ItemsAgent follows a similar path-following technique as the MedianAgent but introduces additional logic to avoid items. Its primary objective is to minimize item collisions while maintaining a stable driving trajectory. The agent leverages peripheral vision detection and movement adjustment strategies to navigate around obstacles.

2 ItemsAgent Algorithm

The ItemsAgent introduces two key algorithms that significantly impact its behavior:

Algorithm 1 Peripheral Vision Check Algorithm

Input: Item position, Kart front direction, Peripheral angle
Output: Boolean indicating if item is within peripheral zone
Compute $direction\_to\_item = \frac{item\_pos}{\|item\_pos\|}$
Compute $kart\_direction = \frac{kart\_front}{\|kart\_front\|}$
Compute angle: $angle = \arccos(clip(direction\_to\_item \cdot kart\_direction, -1.0, 1.0))$
return $angle < peripheral\_angle$

Algorithm 2 CalculateAction Algorithm

Input: Observation data, Lookahead value
Output: Action tuple (steering, acceleration, drift, nitro)
Compute curvature using $compute\_curvature(paths\_end[lookahead - 1])$
Compute slope using $compute\_slope(paths\_end[: lookahead])$
Compute direction to target using $direction\_to\_target = path\_end - kart\_front$
Compute steering: $steering = 0.2 \times direction\_to\_target[0]$
for each item position in $items\_position$ **do**
    if distance_to_item $<$ forecast distance **and** item is in peripheral vision zone **then**
        if item is on the right side **then** $\rightarrow$ Move left aggressively
            if item is on the left side **then** $\rightarrow$ Move right aggressively
            end if

            return $(steering, acceleration, drift, nitro)$

3 Limitations and Future Work

Currently, the ItemsAgent is not functional, and extensive testing has not been performed. Future improvements should focus on refining item detection accuracy, optimizing peripheral vision angle settings, and adjusting movement logic to minimize false positives during item avoidance.

4 Conclusion

The ItemsAgent presents an early-stage implementation of an item-avoidance strategy for SuperTuxKart. While its core logic shows promise, additional testing and refinements are required before achieving stable and competitive racing performance.