# Unraveling the Impact of Reward Functions on Autonomous Racing Performance

Allen Tian, Eddy Guerra, Kecheng Yang

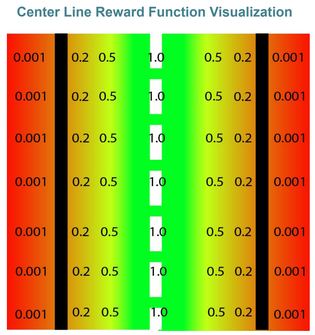
University of Chicago, atian2005@uchicago.edu | University of Houston Downtown, guerrajohne1@gator.uhd.edu | Texas State University, yangk@txstate.edu

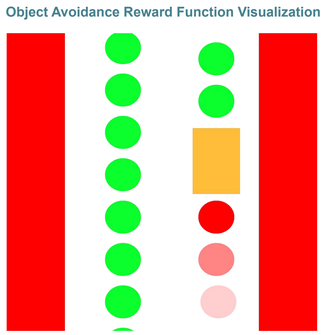
## Introduction

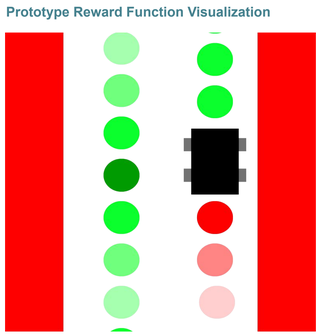
- AWS DeepRacer is a fully autonomous 1/18th scale race car designed to help developers learn and practice reinforcement learning through cloud-based simulations and real-world racing.
- There are 3 modes as of now in the training environment; Time Trial, Object Avoidance, and a newly created Head to Bot.
- A reward function is a way to provide positive or negative feedback to an agent, guiding its learning process in reinforcement learning by assigning numerical values
- Each mode has a reward function that is specially written for it except for Head to Bot because of its novelty.
- Our research focuses on testing the reward functions for Time Trial and Object Avoidance as well as develop a reward function that is suited for the Head to Bot mode.

## Procedure

- Each model(Centerline, Object Avoidance, Prototype) was trained for exactly one hour
- Then we evaluated the models under the Head to Bot evaluation mode which will run the car around the track 10 times. It then saves the metrics and video to a file
- The performance metrics we used were number of crashes, number of off road, and elapsed time.
- We were able to see these metrics for each individual lap, which allowed us to see the full breakdown of what underwent in the evaluation

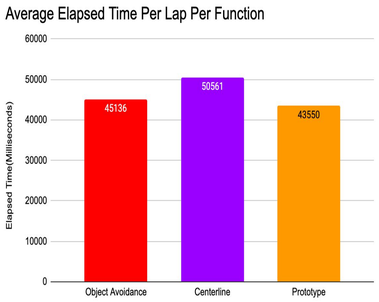


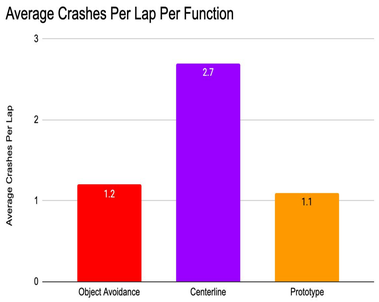



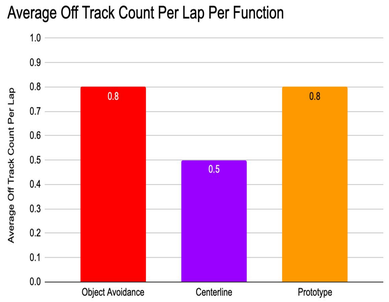




## Reward Functions

- Centerline
  - Assigns rewards based on solely the position of the car in relation to the centerline of the track.
  - Rewards are greatest on the centerline and gradually decrease as it gets closer to the edges of the track
- Object Avoidance
  - Assigns rewards based on the current position of the car in relation to the track and the position of the next object.
  - First calculates reward of whether it is in between the 2 edges of the track.
  - Then calculates whether if the next object is on the path that the bot is traveling on
  - If not on the same path, then give additional reward.
  - If on the same path, then reduce the additional reward based on how close the car is to the next object.
  - Weight and add the rewards
- Our Reward Function(Prototype)
  - Our approach was altering the object avoidance function, as it fares extremely well in the Head to Bot mode.
  - We took everything in the previous reward function, except we added an additional reward for going in the right path
  - Our goal with this was to give it a little extra push to specifically go in an optimal path opposed to going wherever the box/car was not




## Acknowledgements

We would like to thank Texas State University and National Science Foundation. This material is based upon work supported by the National Science Foundation under Grant No. CNS-2149950. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

## Comparison

- For the elapsed time metric, the Prototype model was able to outperform the other two models by a significant margin, yielding average differences of 1586 and 7011 milliseconds with the Object Avoidance model and Centerline model respectively.
- For the crash count metric, the Prototype model was again able to outperform the other two models, yielding average differences of 0.1 and 1.6 crashes per lap with the Object Avoidance model and Centerline model respectively
- For the off track count metric, the Centerline Model was actually able to outperform the Prototype and Object avoidance model by a significant margin, averaging only 0.5 off track counts per lap in contrast with the other two models both averaging 0.8 off track counts per lap.

## Discussion

- As shown in the graphs, the Prototype function performed best or tied for best in 2 of the 3 metrics.
- The Prototype function and Object avoidance function performed relatively similarly, which makes sense as the basis of the Prototype function was the Object avoidance.
- The reinforcement of the optimal path seems to have succeeded as the car decreased the crash count as well as the speed per lap in the Prototype function.
- Centerline took first in off track because the entire premise of Centerline is to follow the center line and gives no regard for crashing into the car.
- Future research could be done in fine tuning of the values of distance in our reward function structure or the weights of each reward in the overall reward function
- Additionally, festing of same functions in Head to Bot mode with longer training times(1.5,2,3 Hours) could also be a good continuation on this research.