PRESENTS

ARTIFICIAL INTELLIGENCE in MOBILITY

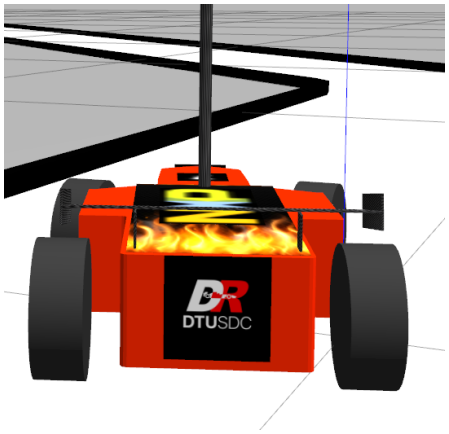
NXP AIM India Online Design Challenge 2021

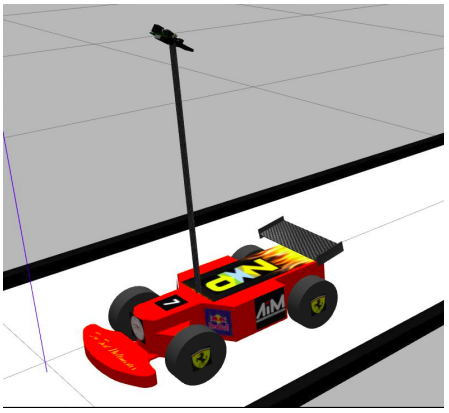
Team Name: DefianzRacing

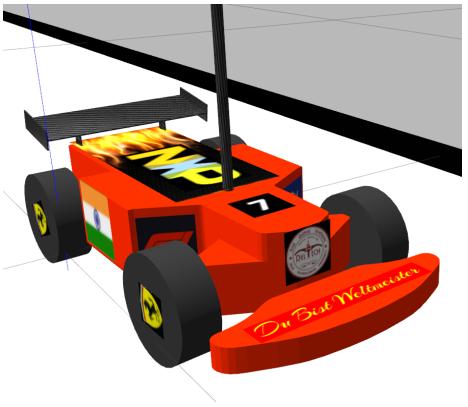
Team Member Details

Name	College/Location	Branch	Semester
Aniket Singh	Delhi Technological University	Computer Engineering	Second
Shaurya Kishore Panwar	Delhi Technological University	Engineering Physics	Second
Harshit Raj	Delhi Technological University	Software Engineering	Second
Tanmay Jain	Delhi Technological University	Computer Engineering	Fourth

NXP AIM CAR SNAPSHOTS:







METHODOLOGY:

Our motion planning algorithm, RRT Star, finds the shortest collision-free path from the starting node to the goal node. Since it has a heuristic to minimize the cost function; and updates the cost function, the resultant trajectory is smooth. This smooth distance-optimized path is the input for the PID controller, and the controller adopts this path. The calculation of the error is by the extent of the deviation from the desired trajectory. There is a constant adjustment of the steering angle by the PID controller to minimize the error. The updated steering angle is passed through a velocity profile algorithm, which we have calculated by using different functions and comparing the overall results. This algorithm takes in steering angle as a parameter and gives the best velocity suited for that steering angle.

ALGORITHM USED AND CHANGES DONE:

MOTION PLANNING - using RRT Star Algorithm

After analyzing different algorithms for motion planning, including ML/AI algorithms we implemented RRT Star, as it gave much better and consistent results. Also, RRT star succeeds in finding the best path, even with dynamic and static objects, making it preferable over other algorithms, as this would be beneficial for the upcoming rounds.

RRT connects points to the nearest available node and the algorithm ends when you reach the goal node. Our motion planning algorithm, RRT Star, is an optimized version of RRT. It finds the shortest collision-free path from the starting node to the goal node. It keeps a record of the distance traveled relative to their parent node. This distance is called the cost function of that node. If a node with a cheaper cost function is present in the fixed radius of the proximal node, it replaces the proximal node. After the addition of a new node, the cost of each node gets updated by its minimum value, which makes the path even more smooth.

PID CONTROLLER

We use the optimized path given by the RRT Star in our PID controller. A PID controller is a control loop feedback mechanism (controller). It calculates an error value as the difference between the current state and the desired state and then tries to minimize it. It achieves in doing so by calculating the steering angle value based on the extent of the error through this formula, which requires tuning for the values of Kp, Kd, Ki.

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt}$$

Velocity Profile Generator - for Maximum Speed

To obtain the best velocity suited for a specific steering angle, we used a unique approach. We derived a formula that gives the maximum velocity when the steering angle is close to zero and vice versa. It helped us achieve our goal of creating a velocity profile generator suited for any track and it can work only by taking the steering angle as a parameter. It enabled us to make our code much more efficient for global use.

$$velocity = k + \frac{e^{-\gamma * angle}}{(1+e^{-\gamma * angle})^2}$$ where k, γ are two parameters

The current velocity of the vehicle is also taken into consideration as it affects the value of the final steering angle. As in the real world, at higher speed, lesser steering input is required to correct the error and vice versa.

$$Final\ Steering = \frac{Raw\ Steering}{(k + currVel)} \quad ; where\ k\ is\ some\ constant$$

Changes Made In Setup

We implemented RRT*(star) algorithm in a separate file called “rrt_star.py” for better readability and code encapsulation. We imported this file using python to use in “aim_line_follow.py,” where we implemented our own algorithms.

We also did the required cosmetic changes to the design of the car in the nxp_cupcar folder. We changed the car color, added the necessary logos, and added few unique details to make the car much more visually appealing.