IEEE Control Systems Letters paper presented at
2023 American Control Conference (ACC)
San Diego, CA, USA. May 31 - June 2, 2023

Monte Carlo Tree Search Based Trajectory Generation for Automated Vehicles in Interactive Traffic Environments

Tinu Vellamattathil Baby*[1] and Baisravan HomChaudhuri[1]

Abstract— **This paper focuses on the development of a trajectory planning method for connected and automated vehicles (CAVs) that takes into account the interactive nature of the vehicles. The proposed approach is based on Monte Carlo Tree Search (MCTS) that traverse through possible actions from each state of the system to identify the trajectory with highest reward. Here, the trajectory is planned and the actions of surrounding vehicles are predicted jointly. Planning the trajectory and predicting the surrounding vehicles jointly in an interactive environment can result in a large action-space, which is not computationally tractable. Hence, we propose an adaptive action-space, which includes pruning the action-space so that the actions resulting in unsafe trajectories are eliminated. The simulation studies show that the proposed approach is capable of identifying less conservative yet safe trajectories for CAVs in a multi-vehicle environment.**

I. INTRODUCTION

Vehicle automation and connectivity is expected to address several issues of the current transportation system. However, full (hundred percent) market penetration of automated vehicles (AVs) or connected and automated vehicles (CAVs) cannot be expected in the near future. Hence, these AVs and CAVs will have to share the road with human-driven vehicles (HDVs), which are rational agents that interact with the AVs and CAVs. CAVs or AVs are generally controlled in a receding horizon fashion, which allow them to utilize information of the surrounding environment. Generally, the overall decision-making architecture involves a high-level sampling-based trajectory planner and a low-level controller that follows the high-level waypoints [1]. Numerous research [2]–[7] have aimed CAV controller development considering full market penetration. Since these works consider all the vehicles to be AVs or CAVs, they can assume that the positions and velocities of the vehicles surrounding the ego AV or CAV is fully available to it, which is not a valid assumption in the presence of HDVs.

Several previous works [8]–[10] that consider interaction between HDVs and CAVs generally only evaluate the impact of the HDVs on the fuel economy. Most of the CAV control methods separate the surrounding vehicle prediction (or future trajectory information obtained via V2V communication from surrounding CAVs) and ego vehicle control problem. This can lead to highly conservative solutions. Since the action of one vehicle influences the others, surrounding vehicle prediction and the ego vehicle controller synthesis problem cannot be treated in an isolated fashion. Inverse Reinforcement Learning-based methods [11] take into account this interactive nature in ego AV controller development, but they are not scalable and can violate safety constraints. Authors in [12] have developed a Monte Carlo Tree Search (MCTS)-based lane-change algorithm for heterogeneous environment, while factoring in the interactive nature of HDVs. However, [12] considered a fixed action-space, which can lead to growing the tree in infeasible directions, and didn't consider a joint state- or action-space.

Vehicles on road constitute an interacting environment and actions of one vehicle affect the future actions of its surrounding vehicles. Hence, a trajectory generation approach in joint state-space and joint action-space consisting the state- and action-space of all the interacting agents might be beneficial. In a predictive control framework, it is essential to incorporate the reactions of the surrounding vehicles to the future actions of the ego vehicle over the prediction horizon. This is essential, as ignoring the reactions may result in identifying the safe maneuvers with higher rewards as unsafe. In this paper, we use MCTS as the trajectory generation framework that searches through the possible actions to identify the best action (control). However, using a joint action-space exponentially increases the number of possible actions as the number of agents increases. When MCTS is implemented in real time, it may not be able to search all the possible actions of the joint system. In such situations, the tree may identify suboptimal solutions, while the control actions with higher rewards can be left unexplored. Hence, we consider an adaptive action-space, where we prune the unsafe actions of each node, thus reducing the size of the joint action-space. We also consider the CAV yaw rate (one of the controls) to be a function of its position at a node, so that infeasible yaw rates are eliminated, thus reducing the size of the action-space. Thus, the paper contributions include (i) developing a MCTS-based trajectory generation framework for an interactive heterogeneous vehicular system that aims at generating non-conservative actions, (ii) developing a technique to adaptively adjust the action-space in MCTS by pruning infeasible actions from each node, thus enabling MCTS to identify safe solutions with higher rewards in a computationally efficient manner, and (iii) evaluating the utility of the framework in a heterogeneous multi-vehicle system and a multi-CAV simultaneous lane-change problem.

This paper is organized as follows: Section II has the

*Address all correspondence to this author. tvellamattathilbaby@hawk.iit.edu

[1]Tinu Vellamattathil Baby (PIN: 154047) and Baisravan HomChaudhuri (PIN: 75966) are with the Mechanical and Aerospace Engineering Department of Illinois Institute of Technology, Chicago, IL 60616, USA

979-8-3503-2806-6/$31.00 ©2023 AACC

mathematical formulation of the problem considered in this paper. Section III details the MCTS-based approach considered. Section IV and V consist of the simulation results and conclusion of the paper.

II. Problem Description

A. System Model

In this work, the motion of a vehicle in a multi-lane road is defined by its four states: longitudinal position s_x, lateral position s_y, velocity v and yaw angle θ. Also, we define the control inputs for a CAV to be its acceleration a and its yaw rate ω. The states are related to the controls and the sampling time T_s by

$$s_x(k+1) = s_x(k) + T_s v(k)\cos(\theta(k)) \tag{1a}$$
$$s_y(k+1) = s_y(k) + T_s v(k)\sin(\theta(k)) \tag{1b}$$
$$v(k+1) = v(k) + T_s a(k) \tag{1c}$$
$$\theta(k+1) = \theta(k) + T_s \omega(k) \tag{1d}$$

B. Optimal Control Problem

Given the above system dynamics, the optimal control problem that a CAV will need to solve is given by

$$\min_{a} \sum_{\tau=k}^{k+T-1} \frac{w_{ocp1}\dot{m}_f(\tau)}{v(\tau)} + w_{ocp2} J(x(\tau), a(\tau)) \tag{2a}$$
$$\text{subject to system dynamics in (1)} \tag{2b}$$
$$s(\tau) \in \mathcal{S}_{safe}(\tau);\ \mathcal{S}_{safe}(\tau) = \mathcal{S}(\tau) \ominus \sum_{j\in\mathcal{N}} \mathcal{S}_j(\tau) \tag{2c}$$
$$v_{min} \le v(\tau) \le v_{max}, \quad a_{min} \le a(\tau) \le a_{max} \tag{2d}$$

where it tries to minimize a cost in (2a) while ensuring satisfaction of system constraints, including the collision avoidance constraint in (2c). The cost in (2a) includes fuel consumption per unit distance ($\dot{m}_f$ is the rate of fuel consumption and w_{ocp1} is its associated weight) and any other cost J (such as travel time) with weight w_{ocp2}. The rate of fuel consumption is modeled as a function of velocity and vehicle acceleration as

$$\begin{aligned}\dot{m}_f(v,a) = {} & 0.5826 + 0.05113v - 0.08799a - 0.00211v^2 \\ & + 0.1565va + 0.02387a^2 + 7.975\times10^{-5}v^3 \\ & - 0.001037v^2a + 0.0465va^2 + 0.02267a^3\end{aligned} \tag{3}$$

where the instantaneous rate of fuel consumption data (obtained from CAN signals) of Cadillac CT6 test vehicle (available at [13]), collected from on-road experiments, is used to model $\dot{m}_f(v,a)$. In (2c), $s(\tau) = [s_x, s_y]^\top$ is the position vector of the CAV, $\mathcal{S}(\tau)$ is the feasible driving region (defined by the roads), $\mathcal{S}_j(\tau)$ is the set of positions occupied by object (vehicle/obstacle) j at τ, and $\mathcal{N}$ is the set of objects (vehicles/obstacles) surrounding the ego CAV. The set $\mathcal{S}_{safe}(\tau)$ in (2c) describes the safe regions where the CAV can be at τ. v_{min} and v_{max} represent the minimum and maximum allowable velocities, whereas a_{min} and a_{max} represent the minimum and maximum allowable accelerations. To solve the above problem in (2) over a horizon T, the CAV needs to predict the future positions of the surrounding vehicles with any prediction algorithm.

III. Trajectory Planning Approach

In this paper, we focus on developing control strategies for CAVs. The CAV control problem in (2) is generally split into a trajectory generation problem that generates the reference trajectory of the vehicle, followed by a motion control problem that tracks the reference trajectory in an optimal manner. To this end, we present a MCTS-based approach that considers the joint state-space and action-space for all the interacting agents in a ego CAV's surrounding, and predicts the future actions of the surrounding vehicles while taking into account the ego CAV's actions over the horizon.

MCTS is a search algorithm that combines the classical tree search and reinforcement learning. The classical tree search strategy keeps exploring the current best action, which grows the tree in depth. The reinforcement learning aspect of the strategy explores other actions periodically, which grows the tree in breadth. MCTS is generally characterized by the exploration-exploitation trade-off. This helps MCTS to not only explore the current best actions further in time, but also the other actions that can potentially provide better reward. This trade-off is implemented by the upper confidence bounds for trees (UCT) [14] given by

$$UCT = \frac{r_{a_c}}{n_{a_c}} + c\sqrt{\frac{\log N}{n_{a_c}}} \tag{4}$$

where r_{a_c} is the reward associated with implementing action a_c and n_{a_c} is the total number of times action a_c is selected. N is the total number of simulations of MCTS and c is a constant that facilitates the trade-off between exploration and exploitation. In MCTS, each state of the system represents a node, and each action determine the transition from one node to another. Growing the tree by adding nodes is a four step process that involves *selection*, *expansion*, *simulation* and *backpropagation*. A brief description of each process is presented in this paper. For more details, please refer to [15]–[17].

1) Selection: In this process, the current tree is traversed from root node (the initial state or current state) to leaf node (node with unexplored child nodes). Here, the UCT value is evaluated for each node, and the node with the largest value is chosen as the best node. This process is continued until a leaf node is reached, which is then expanded.
2) Expansion: An unexplored node (obtained as a result of performing an unexplored action) is chosen randomly and is added as a child of the node that was identified during the selection process.

3) Simulation: In this process, a simulation is performed starting from the child node (added during the expansion phase) with a preassigned simulation policy. Generally, the preassigned simulation policy includes choosing random actions till the end of the prediction horizon. Then, the reward is calculated for the child node based on the simulated actions.
4) Backpropagation: The reward calculated for the child node (in the simulation phase) is propagated backwards through parent nodes to the root node. Also, the number of visits for each parent node is incremented.

In this paper, the reward r_{a_c} calculated during simulation process is based on (5a), where Δs represents the longitudinal distance travelled over the horizon and $F = \sum_\tau T_s \dot{m}_f(\tau)$ represents the fuel consumed to travel Δs. The reward function in (5a) maximizes the distance travelled by using unit amount of fuel (i.e., maximizes miles per gallon). The reward for collision is expressed in (5b), which is taken to be a large negative value, so that actions resulting in collisions are avoided with an associated weight of w_1. (5c) represents the reward for travelling close to the desired velocity (v_{des}) with an associated weight of w_2. (5a) also includes penalty terms to avoid large accelerations or braking with weight w_3.

$$r_{a_c} = \frac{\Delta s}{F} + w_1 r_c + \sum_\tau (w_2 r_v(\tau) - w_3 a^2(\tau)) \quad (5a)$$

$$r_c = \begin{cases} 1 & \text{if no collision} \\ -10^{10} & \text{if collision occurs} \end{cases} \quad (5b)$$

$$r_v(\tau) = \begin{cases} 1 & \text{if } v_{diff}(\tau) \le 1 \\ 1 - \frac{v_{diff}(\tau)}{v_{des}(\tau)} & \text{if } v_{diff}(\tau) > 1 \text{ and} \\ & v_{diff}(\tau) \le v_{des}(\tau) \end{cases} \quad (5c)$$

with $v_{diff}(\tau) = |v(\tau) - v_{des}(\tau)|$.

A. Original action-space

The surrounding human-driven vehicles (HDVs) are modeled with a car-following model, such as the Gipps' car following model ([18]), for longitudinal control. HDVs are considered to change lanes when the average velocity of its adjacent lane is higher than its own velocity. The HDVs perform safety checks derived from Gipps' car following model and Intelligent Driver Model (IDM) [19] before initiating any lane change.

We consider the control inputs of the connected and automated vehicle (CAV) to be its acceleration a (m/s^2) and its yaw rate ω (rad/s). In [12], the action-space (acceleration and yaw-rate) was discretized into 14 combinations. The vehicle acceleration was discretized from -3.5 m/s^2 to 2.5 m/s^2, while the yaw rate was discretized with a step size of $\frac{\pi}{4}$ rad/s from $-\frac{\pi}{2}$ rad/s to $\frac{\pi}{2}$ rad/s. Although action-space discretization reduces solution optimality, reducing the discretization step size (hence increasing the action-space) increases the computational complexity to a large extent. In this paper, the original action-space $\mathcal{A}$ is taken as 14 combinations of the two controls as shown below. However, unlike [12], our yaw rate actions are considered

$$\begin{bmatrix} 2.5 & 1.5 & 0 & -1.5 & -3.5 & -5 & 0 & 0 & 1 & 1 & -1 & -1 & -3.5 & -3.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & \omega & -\omega & \omega & -\omega & \omega & -\omega & \omega & -\omega \end{bmatrix}^T$$

Fig. 1: The original action-space for the CAV performing a MCTS.

to be a function of the CAV's lateral position and intended lane-change action to avoid infeasible solutions. The first row of the above matrix represents the acceleration (in m/s^2) and the second row represents the yaw rate (rad/s), which is calculated during the pruning of action-space (in subsection III-B). Although there are 14 possible actions considered, a few of them can lead to infeasible solutions, hence, growing the nodes from those actions is unnecessary. We thus propose pruning of the action-space, so that actions leading to infeasible nodes are eliminated early on.

B. Pruning of action-space

MCTS method randomly explores through the possible actions to identify its best child node (next action). For realistic scenarios, the computation time should be less than the sampling time, and a small sampling time in vehicular applications can result in the action-space to be explored partially. When the action-space is large, only a small part of the solution-space gets explored within the stipulated time. As a result, most or even all the safe nodes (that avoid collision) can be left unexplored, leading to infeasible solutions. Hence, we propose to prune the action-space of the algorithm depending on the node of the tree to remove the unsafe/undesired actions, thus reducing the size of the action-space. The pruning of action-space is performed in two steps, i) eliminating the undesired lane-change actions, and ii) eliminating the undesired accelerations. These techniques are described below.

1) Pruning of lane-change actions: Pruning of lane-change actions reduces the angular rate (yaw rate) action-space from a given node. The pruning is performed by identifying the possible 'target lane(s)' for the CAV. Assuming that by the end of the decision-making horizon, the vehicle would either stay in its lane or move to one of the adjacent lanes, the target lanes include the vehicle's current lane and the lanes to its immediate left/right. We consider high-level lane-change actions to be denoted as $\ell = \{-1, 0, 1\}$, where 0, -1, and 1 refer to 'remain in lane,' 'change lane to the right,' and 'change lane to the left,' respectively. For each of the high-level actions, only a subset of the action-space (discussed in section III-A) is valid. The yaw-rate action-space is dependent on these high-level lane-change actions ℓ. The possible target lanes and the corresponding pruned lane-change actions are as follows:

1) When the ego CAV's yaw angle θ in a node is less than 0.002 radians and there is no object (vehicle/obstacle)

less than 100 m ahead in the same lane, then its target lane is the vehicle's current lane and the only action possible is 'remain in lane'. In this case, the pruned possible lane-change actions will be $\ell = \{0\}$.

2) When the CAV's yaw angle θ in a node is less than 0.002 radians and there are objects (vehicle/obstacle) less than 100 m ahead in the same lane, then its target lanes can be either its current lane defined by the action 'remain in lane' or one of its adjacent lanes defined by the action 'change-lane'. In this case, the pruned lane-change actions will be $\ell = \{0, 1\}$ or $\{0, -1\}$.
3) When the CAV's yaw angle θ in a node is higher than a threshold, then its target lane is the one identified at previous instants (that initiated lane-change), and it continues lane-change action which is defined by either $\ell = \{1\}$ or $\{-1\}$.

For each of the cases discussed above, the actions space $\mathcal{A}(\ell)$ will be a subspace of the original action-space shown in Fig. 1. For case 2 above, the target lane q (immediate left/right or same lane) is identified as the lane with maximum relative distance d_{R_q}, i.e., $q = \arg\max_q \ d_{R_q}$, with

$$d_{R_q}(k) = d_{m_q}(k) - d_{min} + T_n(v_{q500}(k) - v(k)) \qquad (6)$$

$q \in \{lt, sm, rt\}$ represents the left, same, and right lanes. d_{m_q} and v_{q500} represents the distance to the nearest object and the average velocity of all vehicles travelling ahead within 500 m in the q^{th} lane, respectively. d_{min} is a predefined safety distance and $v(k)$ is the velocity of the CAV. T_n is a parameter similar to headway time.

Calculating yaw rate: Once the high-level action-space from a given node has been identified as a subset of $\{-1, 0, 1\}$, the corresponding feasible yaw rate(s) is then calculated. When the pruned action-space includes $\ell = 0$, then the yaw rate (the second row in Fig. 1) is calculated as $\omega(k) = -\theta(k)/T_s$ to make the vehicle heading angle θ zero. When the pruned action-space includes 'change-lane' ($\ell = 1$ or -1) the yaw rate is calculated from the lateral distance Δy that needs to be traversed during the next sampling time T_s. Δy is taken to be $\frac{T_s L_w}{T_\ell}$, where L_w is the lane width and the lane-change is assumed to be completed in T_ℓ seconds. The yaw rate is then calculated as

$$|\omega(\tau)| = \frac{1}{T_s}\left(\sin^{-1}\left(\frac{\Delta y}{T_s v(\tau)}\right) - \theta(\tau)\right) \qquad (7)$$

The sign of ω is positive if the target lane is the left lane and is negative if it is the right lane. We initially assume $T_\ell = 3$ secs. We also constraint $|\omega(\tau)|$ to ensure the vehicle stays in the road, i.e., $|\omega(\tau)| \leq \omega_{max}(s_y)$, so that $|\omega(\tau)| = \min(|\omega(\tau)|, \omega_{max}(s_y))$, where $\omega_{max}(s_y)$ is dependent on the lateral position of the vehicle.

2) Pruning of acceleration space: The maximum safe velocity (v_{safe}) is calculated to identify the maximum safe acceleration, that is used to eliminate the undesired/unsafe accelerations. The maximum safe velocity, v_{safe}, is defined as the maximum velocity at which a vehicle can travel without colliding with other vehicles/stationary obstacles, and is calculated as

$$v_{safe}(k) = \frac{d_{sm}(k) - d_{min} + \eta T_s w_4 v_{sm}(k)}{\eta T_s} \qquad (8)$$

where the distance $d_{sm}(k)$ is the distance to the nearest object (vehicle/obstacle) ahead of the CAV in the same lane, and $v_{sm}(k)$ is its velocity. w_4 is an adaptive weight (for this paper, $w_4 \in [0.8, 1]$) added to incorporate any decrease in v_{sm} that can result in collision. As $v_{sm}(k)$ increases from 0 to v_{max}, w_4 decreases from 1 to 0.8, where v_{max} is the road speed limit. Similar to headway time, η is added as a tolerance for time. The maximum acceleration (from the original action-space) should be less than or equal to $(v_{safe}(k) - v(k))/T_s$ and hence all values greater than $(v_{safe}(k) - v(k))/T_s$ are eliminated (pruned). By adaptive pruning of the action-space in Fig. 1 based on the node of the tree, we only grow the tree in the feasible direction that can provide potentially higher reward.

Once the high level MCTS based motion planner identifies the reference trajectory x_{ref}, the low-level MPC based motion controller solves the problem in (2) with $J = ||x - x_{ref}||^2$ to identify fuel efficient control inputs.

IV. Simulation Results

To evaluate the performance of our proposed method, we consider two cases: (i) a single CAV surrounded with HDVs in a congested traffic (e.g., due to road construction), and (ii) a two CAV simultaneous lane-change problem. First, we considered a scenario consisting of four HDVs and one CAV. The vehicles are considered to travel on a two lane road that have a stationary obstacle (such as construction in progress). The stationary object/obstacle, can also be considered to capture a very slow moving traffic in a lane. The initial positions and velocities of the vehicles, as well as the position of the obstacle are depicted in Fig. 2. We also consider a road speed limit of 45 mph (≈ 20.12 m/s).

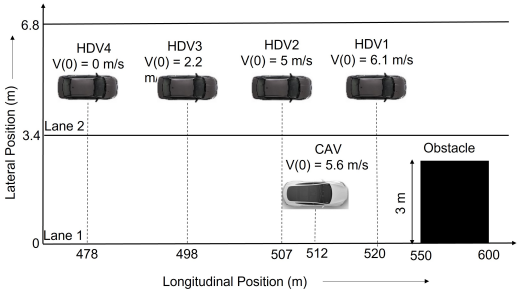
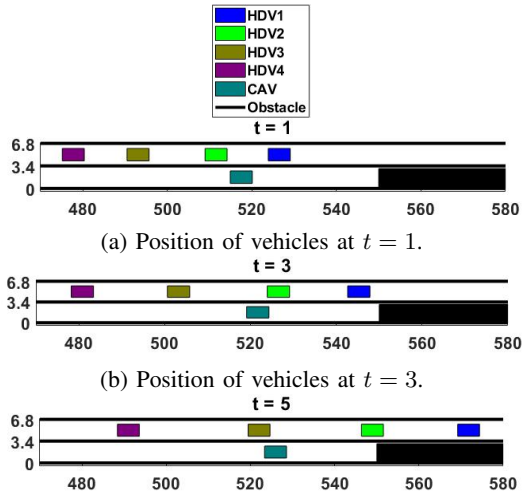


Fig. 2: Illustration of the scenario and initial conditions for simulation.

In order for the CAV to evaluate a fuel efficient trajectory, the CAV should have a prediction model for its surrounding vehicles. In the scenario depicted in Fig. 2, all vehicles surrounding the CAV are HDVs. For comparison purposes, we consider longitudinal acceleration of the HDVs follow

car-following models (e.g., Gipps' model), and the lane-change decisions are made based on Intelligent Driver Model (IDM). Modeling the HDVs with car-following models is common in the literature. Hence, the actions of a HDV is dependent on its preceding, as well as its surrounding vehicles. Since we are planning to develop a receding horizon controller (or model predictive controller) for the CAV, the CAV needs to predict the behaviour of the HDVs over the prediction horizon. Hence, over the model predictive control (MPC) horizon T, the CAV predict the HDVs' actions on the assumption that the HDVs consider the CAV to maintain its action (i.e., maintain its velocity and lane). The positions of the interacting vehicles over the MPC horizon at times $t = 1, 3, 5$ in such a situation is shown in Fig. 3. The CAV ends up maintaining its lane as it cannot finds a safe lane-change maneuver that avoids the predicted HDVs, and has to travel at a very low speed due to the obstacle in its lane.



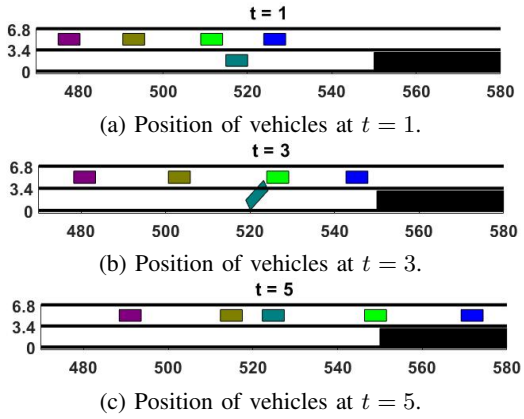
(a) Position of vehicles at $t = 1$.

(b) Position of vehicles at $t = 3$.

(c) Position of vehicles at $t = 5$.

Fig. 3: Position of vehicles over horizon without considering the future reactions of surrounding vehicles to future actions of CAV.

On the other hand, Fig. 4 shows the vehicle trajectories based on our proposed MCTS approach (including the low-level controller), where the ego CAV jointly predicts and plans its trajectory over a horizon T. Here, the CAV takes into account the reaction of the surrounding HDVs to its own future positions (actions), which results into the CAV finding maneuvers (which will be more time and fuel efficient) that allows it to change lane (as shown in Fig. 4) and move at an optimal velocity. In this scenario, the CAV identifies that $HDV3$ will react to its initiation to change lane and slow down (since the HDVs are rational agents), making room for the CAV to complete its lane-change. It follows a mildly aggressive yet safe trajectory that involves lane-change, thus avoiding stopping at the obstacle. If the CAV had modeled the HDVs to be unresponsive to its actions, it will not be able to change its lane as it will detect a collision with $HDV3$ at time $t = 5$, as shown in Fig. 5. The pruning of action-space enabled the proposed approach to find safe solutions in a computationally efficient fashion. With a pruned action-space, the Monte Carlo Tree was expanded over the horizon in 2.23 seconds instead of 40.61 seconds in case of an action-space without pruning, which is 95% more efficient.



(a) Position of vehicles at $t = 1$.

(b) Position of vehicles at $t = 3$.

(c) Position of vehicles at $t = 5$.

Fig. 4: Position of vehicles over horizon considering the future reactions of surrounding vehicles to future actions of CAV.

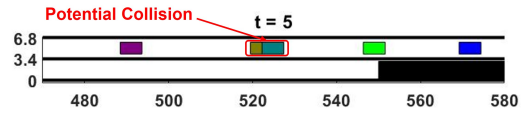


Fig. 5: Position of vehicles showing potential collision if the CAV had chosen lane-change while still assuming its surrounding HDVs to be unresponsive to its actions.

We apply our proposed approach in multi-CAV problem, where two CAVs simultaneously change lanes. This scenario consists of two HDVs and two CAVs travelling on a two lane road, as shown in Fig. 6. We also consider a road speed limit of 45 mph (≈ 20.12 m/s).

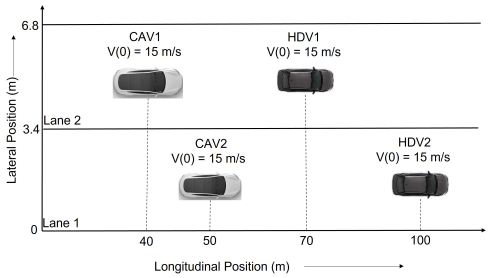
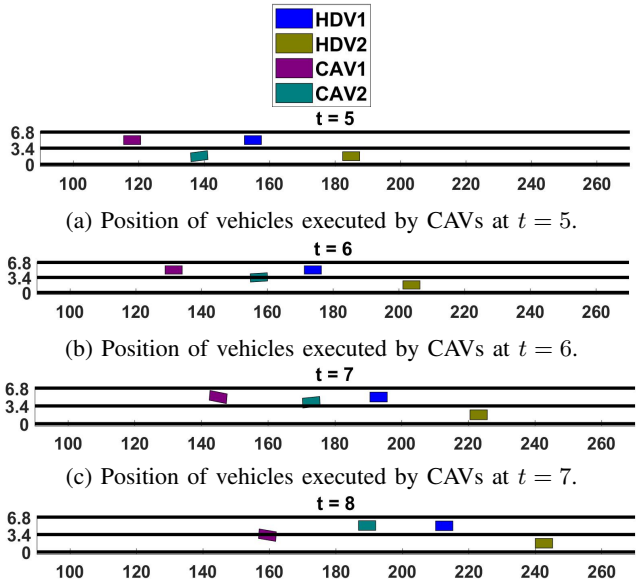


Fig. 6: Illustration of the second scenario and initial conditions for simulation.

Here, there are two CAVs ($CAV1$, $CAV2$), and we assume that both CAVs intend to change their respective lanes. Rather than the CAVs sharing their future state or position trajectory over the horizon (as done in most CAV control problems), we only consider the CAVs to share their lane-change intent. Each CAV assumes that the other CAV, once it communicate its intend to change lane, will initiate the lane-change within next 5 seconds. With this assumption,

both the CAVs perform the proposed MCTS-based trajectory generation approach independently in the joint state-space.

Figure 7 shows the the actions of $CAV1$ and $CAV2$ that are obtained by searching the Monte Carlo Trees of $CAV1$ and $CAV2$, respectively. The figure shows that both the CAVs can change lanes safely based on their predictions of the actions of the other CAV. Thus, the CAVs are able to perform safe simultaneous lane changes without sharing their future trajectory information or needing an iterative approach of trajectory modification.



(a) Position of vehicles executed by CAVs at $t = 5$.

(b) Position of vehicles executed by CAVs at $t = 6$.

(c) Position of vehicles executed by CAVs at $t = 7$.

(d) Position of vehicles executed by CAVs at $t = 8$.

Fig. 7: Position of vehicles over time.

V. Conclusion

We propose a MCTS-based trajectory planning algorithm in a system consisting of interacting agents. Here, a joint state-space and joint action-space is considered that defines the state-space and action-space of the interacting agents. Since the possible actions in a joint action-space exponentially grow with the number of agents, we propose a technique to prune the action-space so that unsafe actions for each agent is eliminated, thus reducing the size of the action-space. Simulation studies show that the proposed approach helps in identifying trajectories that are non-conservative. Future works include incorporation of advanced prediction algorithms with trajectory generation approach.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation (grant #2130718).

References

[1] J. Guanetti, Y. Kim, and F. Borrelli, "Control of connected and automated vehicles: State of the art and future challenges," *Annual Reviews in Control*, 2018.

[2] E. Hellström, J. Åslund, and L. Nielsen, "Design of an efficient algorithm for fuel-optimal look-ahead control," *Control Engineering Practice*, vol. 18, no. 11, pp. 1318–1327, 2010.

[3] B. Asadi and A. Vahidi, "Predictive cruise control: Utilizing upcoming traffic signal information for improving fuel economy and reducing trip time," *IEEE transactions on control systems technology*, vol. 19, no. 3, pp. 707–714, 2011.

[4] A. F. Canosa and B. HomChaudhuri, "Computationally-efficient fuel-economic high-level controller design for a group of connected vehicles in urban roads," in *ASME 2018 Dynamic Systems and Control Conference*, pp. V002T15A002–V002T15A002, American Society of Mechanical Engineers, 2018.

[5] B. HomChaudhuri, A. Vahidi, and P. Pisu, "Fast model predictive control-based fuel efficient control strategy for a group of connected vehicles in urban road conditions," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 760–767, 2017.

[6] B. HomChaudhuri, R. Lin, and P. Pisu, "Hierarchical control strategies for energy management of connected hybrid electric vehicles in urban roads," *Transportation Research Part C: Emerging Technologies*, vol. 62, pp. 70–86, 2016.

[7] B. HomChaudhuri, A. Vahidi, and P. Pisu, "A fuel economic model predictive control strategy for a group of connected vehicles in urban roads," in *2015 American Control Conference*, pp. 2741–2746, 2015.

[8] L. Cui, H. Jiang, B. B. Park, Y.-J. Byon, and J. Hu, "Impact of automated vehicle eco-approach on human-driven vehicles," *IEEE Access*, vol. 6, pp. 62128–62135, 2018.

[9] J. Rios-Torres and A. A. Malikopoulos, "Impact of partial penetrations of connected and automated vehicles on fuel consumption and traffic flow," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 4, pp. 453–462, 2018.

[10] J. Monteil and G. Russo, "On the coexistence of human-driven and automated vehicles within platoon systems," in *18th European Control Conference (ECC), Naples, Italy*, pp. 3173–3178, 2019.

[11] D. Sadigh, N. Landolfi, S. S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for cars that coordinate with people: leveraging effects on human actions for planning and active information gathering over human internal state," *Autonomous Robots*, vol. 42, no. 7, pp. 1405–1426, 2018.

[12] S. Karimi and A. Vahidi, "Receding horizon motion planning for automated lane change and merge using monte carlo tree search and level-k game theory," in *Proceedings of American Control Conference*, pp. 1223–1228, 2020.

[13] "Light-duty connected and automated vehicle functionality in real-world operational scenarios." `https://livewire.energy.gov/project/ld-cav-functionality`, 2022.

[14] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *Machine Learning: ECML 2006*, pp. 282–293, 2006.

[15] S. Gelly and D. Silver, "Monte-carlo tree search and rapid action value estimation in computer go," *Artificial Intelligence*, vol. 175, no. 11, pp. 1856–1875, 2011.

[16] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.

[17] P. I. Cowling, E. J. Powley, and D. Whitehouse, "Information set monte carlo tree search," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 2, pp. 120–143, 2012.

[18] P. Gipps, "A behavioural car-following model for computer simulation," *Transp. Res. Part B Methodological*, vol. 15, pp. 105–111, 1981.

[19] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, pp. 1805–1824, 02 2000.