MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Improved A-Search Guided Tree for Autonomous Trailer Planning

Leu, Jessica; Wang, Yebin; Tomizuka, Masayoshi; Di Cairano, Stefano

TR2022-133 October 25, 2022

Abstract

This paper presents a motion planning strategy that utilizes the improved A-search guided tree to enable autonomous parking of a general 3-trailer with a car-like tractor. Different from the state-of-the-art state-lattice-based methods where numerous motion primitives are necessary to ensure successful planning, our work allows quick off-lattice exploration to find a solution. Our treatment brings at least three advantages: fewer and lower design complexity of motion primitives, improved success rate, and increased path quality. Unlike on-lattice exploration, where the cost-to-go is obtained by querying a heuristic look-up table, off-lattice exploration entails the heuristic function being well-defined at off-lattice nodes. We train a neural network through reinforcement learning to model the maneuver costs of the trailer and use it as the heuristic value to better approximate the cost-to- go. Simulations demonstrate the effectiveness of the proposed method in terms of planning speed and path length.

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2022

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Improved A-Search Guided Tree for Autonomous Trailer Planning

Jessica Leu, Yebin Wang, Masayoshi Tomizuka, and Stefano Di Cairano

Abstract— **This paper presents a motion planning strategy that utilizes the improved A-search guided tree to enable autonomous parking of a general 3-trailer with a car-like tractor. Different from the state-of-the-art state-lattice-based methods where numerous motion primitives are necessary to ensure successful planning, our work allows quick off-lattice exploration to find a solution. Our treatment brings at least three advantages: fewer and lower design complexity of motion primitives, improved success rate, and increased path quality. Unlike on-lattice exploration, where the cost-to-go is obtained by querying a heuristic look-up table, off-lattice exploration entails the heuristic function being well-defined at off-lattice nodes. We train a neural network through reinforcement learning to model the maneuver costs of the trailer and use it as the heuristic value to better approximate the cost-to-go. Simulations demonstrate the effectiveness of the proposed method in terms of planning speed and path length.**

I. Introduction

Autonomous tractor-trailer systems have attracted strong interests from both industry and academia due to their high cargo transportation efficiency. However, their complicated kinematics pose significant challenges in both control and planning, particularly, when reversing maneuvers and collision avoidance are needed.

Early works proposed flatness-based trajectory generation for unconstrained environments [1], [2], while others proposed hierarchical planners [3], [4] that first plan a collision-free holonomic path and then iteratively modify it to a kinematically feasible trajectory. Recent works resorted to the state-lattice framework to accomplish kinodynamic planning [5]–[7], where the dynamic feasibility and collision avoidance are addressed simultaneously. These algorithms search a graph, where the vertices are discrete states and the edges are from a set of precomputed motion primitives (MPs). Since the MPs can be generated offline by solving optimal control problems (OCP), the difficulty incurred by the dynamics is handled offline. Resolution optimality and completeness are guaranteed by state-lattice-based planners.

A major limitation with state-lattice-based methods is the curse of dimensionality. Works [7], [8] restricted the MPs to those admitting transition between circular equilibrium configurations. This lowers the dimension of the search space that planners explore. However, a large amount of MPs is necessary. The trade-off between the number of MPs and planning success rate/planning accuracy needs to be managed carefully so that the planner can find a path that ends close to the goal within a reasonable time. A well-informed heuristic function that correctly approximates the true cost-to-go from a state to the goal state is needed to maintain real-time performance [5], [9]. Work [9] combines Euclidean distance and a free-space heuristic look-up table (HLUT) to calculate the heuristic value. Nevertheless, the memory burden of storing the HLUT may be of concern for certain applications, which could be as large as hundreds of MB memory.

A remedy to lattice-based limitations is an improved A-search guided tree (i-AGT) [10], which is constructed on-the-fly and accepts off-lattice exploration. This work adopts and extends i-AGT to tractor-trailer systems since moving away from state-lattice-based methods enables using a smaller set of MPs with similar planning success rate and yields better path quality, particularly, shorter path length. We propose a construction process to generate the set of MPs for off-lattice use and analyze its connectivity to achieve reasonable performance. On top of these advantages, i-AGT groups MPs into various modes with their associated priorities, allowing a mode selection process to improve computation efficiency of node expansion. Unlike lattice-based methods where the cost-to-go can be readily constructed by as the HLUT, i-AGT with off-lattice nodes requires a more sophisticated design to estimate the cost-to-go. We reckon that the cost-to-go oftentimes largely depends on the "level of maneuver difficulty". In this light, we learn the maneuver costs of the complicated trailer kinematics by reinforcement learning, and use the learned value function to obtain the heuristic value.

Thus, this work presents a motion planning method that utilizes i-AGT with a data driven heuristic to plan for autonomous tractor-trailer systems. Main contributions are:

- A simple set of MPs is generated for i-AGT with reachability guarantees.
- A data-driven heuristic is proposed to increase search efficiency of i-AGT.
- Extensive simulation is performed to show the effectiveness of the proposed system.

II. Related Works and Preliminaries

A. Related Works

Sampling-based motion planning, such as RRT [11] and PRM [12], is popular for its computation efficiency and ease of implementation. On the other hand, optimization-based algorithms [13]–[15] require a non-trivial initialization to converge, especially in cluttered environments [16], [17],

J. Leu and M. Tomizuka are with with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720, USA. This work was done while she was a research intern at Mitsubishi Electric Research Laboratories. (email: {jess.leu24,tomizuka}@berkeley.edu).

Y. Wang and S. Di Cairano are with Mitsubishi Electric Research Laboratories, Cambridge, MA 02139, USA (email: {yebinwang,dicairano}@ieee.org).

[17], [18]. Search-based planning is another popular method for tractor-trailer systems, which abstracts the configuration space as a graph with nodes and edges [7], [19]. Search operations are often done by A* [6], [20], [21]. State-lattice-based planners are deterministic sampling-based planners which uses a finite set of precomputed MPs online to find a resolution-optimal solution [9], [22]. However, due to the discretized search space, the graph resolution will affect optimality [23].

B. Problem Statement

Consider a system with the following dynamics

$$\dot{X} = f(X) + g(X, u), \tag{1}$$

where $X \in \mathcal{X} \subset \mathbb{R}^{n_x}$ is the state, $u \in \mathcal{U} \subset \mathbb{R}^m$ is the control input, f is a smooth vector field, and $g = (g_1^\top, \cdots, g_m^\top)^\top$, where g_i is a smooth vector field. A *configuration* of system (1) is a complete specification of the position of every points of that system. The *configuration space* $\mathcal{C} \subset \mathbb{R}^{n_c}$ is a compact set representing all possible configurations; and $\mathcal{C}_{free}$ denotes a collision-free configuration space. This work assumes $\mathcal{C} = \mathcal{X}$ and the motion planning problem as follows:

Problem 2.1: Given an initial configuration $X_0 \in \mathcal{C}_{free}$, a goal configuration $X_f \in \mathcal{C}_{free}$, and system (1), find a feasible trajectory $\mathcal{P}_t$ which

(I) starts at X_0 and ends at X_f, while satisfying (1); and
(II) lies in the collision-free configuration space $\mathcal{C}_{free}$.

C. Trailer Modeling and control

Consider a front wheel drive *standard trailer system* [24], [25] as shown in Fig. 1, where $(x, y)^\top$ are the coordinates of the midpoint of the tractor's rear wheel axis, θ_0 is the tractor orientation, $\theta_1, \theta_2, \theta_3$ are the orientations of trailers, v_f is the front-wheel velocity of the tractor, δ is the steering angle of the tractor, and L is the distance between (x, y) and the midpoint of the front wheel axis. The control inputs are v_f and δ. A mechanical constraint $|\delta| \leq \delta_{\max}$ limits the minimum turning radius R of a path. Provided that v_f and δ can be independently controlled, we introduce new control variables: $u = (v, s)^\top = (\cos(\delta)v_f, \frac{\tan(\delta)}{\tan(\delta_{\max})})^\top$, where $\tan(\delta_{\max}) = \frac{L}{R}$, and $s \in [-1, 1]$ is the normalized steering angle. It is beneficial to represent the kinematic model in the coordinates $\xi = (x, y, \theta_0, \theta_1 - \theta_0, \theta_2 - \theta_1, \theta_3 - \theta_2)^\top$, where:

$$\begin{aligned}
\dot{x} &= \cos(\theta_0)v \\
\dot{y} &= \sin(\theta_0)v \\
\dot{\theta}_0 &= \frac{vs}{R} \\
\dot{\xi}_4 &= -v\frac{d_1 s + \sin(\xi_4)R}{Rd_1} \\
\dot{\xi}_5 &= -v\frac{d_1\cos(\xi_4)\sin(\xi_5) - d_2\sin(\xi_4)}{d_1 d_2} \\
\dot{\xi}_6 &= -v\cos(\xi_4)\frac{d_2\cos(\xi_5)\sin(\xi_6) - d_3\sin(\xi_5)}{d_2 d_3}.
\end{aligned} \tag{2}$$

In ξ-coordinates, the constraints to prevent a jack-knife configuration are

$$|\xi_i| \leq \xi_{\max}, \quad 4 \leq i \leq 6, \tag{3}$$

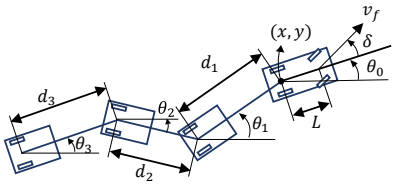


Fig. 1: Kinematics of a front drive tractor with 3 trailers. (All trailers are on-axle and all angles representing the orientation of tractor and trailers ($\theta_i, i = 0, \ldots, 3$) are w.r.t. the x-axis.)

where $\xi_{\max}$ must be less than $\frac{\pi}{2}$. The trailer system is subject to additional state and control constraints:

$$0 \leq \theta_0 < 2\pi, \ |v| \leq v_{\max}, \ |s| \leq 1. \tag{4}$$

Remark 2.2: As manifested below, this work can be readily generalized to other trailer systems.

To check for collisions, we wrap around the tractor, the trailers, and the obstacles with rectangles. A controller based on model predictive control [26] can control the trailer to follow a reference trajectory.

D. i-AGT Algorithm

i-AGT constructs a tree $\mathcal{T}$, with a root node $X_{init} \in \{X_0, X_f\}$ and a goal $X_{goal} \in \{X_0, X_f\} \backslash \{X_{init}\}$, which reaches a neighbor $\mathcal{B}_\epsilon(X_{goal}) \triangleq \{X \in \mathcal{X} | d(X, X_{goal}) \leq \epsilon\}$. Specifically, $d(\cdot, \cdot)$ is a distance function, e.g. a weighted 2-norm: $\|X_i - X_j\|_P = ((X_i - X_j)^\top P(X_i - X_j))^{1/2}, X_i, X_j \in \mathcal{C}$. Similar to A*, each node X is assigned an F-value calculated as follows:

$$F(X) = g(X_{init}, X) + h(X, X_{goal}), \tag{5}$$

where $g(X_{init}, X)$ represents the cost-to-come, or the arrival cost, from X_{init} to X, and $h(X, X_{goal})$ denotes the estimated cost-to-go, or the heuristic value, from X to X_{goal}. i-AGT maintains a priority queue Q, which contains nodes to be expanded. All nodes in Q are ordered according to their F-values.

The trade off between planning time and maneuver resolution, i.e., the cardinality of the set of MPs, denoted by $\mathcal{M}$, has to be made [23]. i-AGT pivots on a concept 'mode' which divides $\mathcal{M}$ into m subsets of motion primitives $M_i \subset \mathcal{M}, i = 1, \ldots, m$. During node expansion of the best candidate node X_{best}, a current mode M_c which has the highest priority among untried modes is determined. Then, X_{best} is expanded by applying primitives in M_c one by one. Applying each MP gives a new node X_k and the connecting trajectory $\mathcal{P}_k$ from X_{best} to X_k. If X_k is δ-distant away from $\mathcal{T}$ and $\mathcal{P}_k$ is collision free, then the algorithm

(I) updates priority $P^{M_c}_{X_{\text{best}}}$ according to $F(X_{\text{best}})$ and $F(X_k)$;
(II) copies the mode priorities of X_{best} to X_k;
(III) adds node X_k and edge $E(X_{\text{best}}, X_k)$ to $\mathcal{T}$;
(IV) inserts node X_k into Q.

Details of i-AGT can be found in [10].

III. METHODS

i-AGT entails three components: the MPs which governs node expansion, the mode priority which governs mode selection, and the heuristics which governs node selection. A naive approach is to directly apply the i-AGT used in [10] to the trailer planning setting, i.e., use the same heuristics and mode definition, while defining the MPs as a tuple of constant velocity and constant steering angle over a certain period of time. In Case 7 (Fig. 5(f)), the i-AGT constructs a tree with 2617 nodes using 7sec. However, it undergoes heavy computation, because the MPs result in exploring the 6-dim state space and land too many collision-free but kinematically undesirable nodes. Planning over the 6-dim state space is dominated by the curse of dimensionality.

The following sections introduce our proposed methods to enable trailer planning using i-AGT, particularly, the construction of the MPs set and obtaining a informative heuristics for a trailer system.

A. Dimension Reduction

To find a better set of MPs, we follow the well-established idea in [9], [25] to circumvent the curse of dimensionality by restricting X to meet the condition that the tractor and all trailers move in circles (trailers and tractor have the same yaw rate C). Given v, the yaw rate of the tractor and the headings of all trailers are uniquely determined by the steering action s. For tractor $\dot{\theta}_0 = vs/R = C(s)$, the headings of trailers can be uniquely determined from $\dot{\xi}_k = 0, \quad k = \{4, 5, 6\}$, which admit the solutions

$$\begin{aligned} &\xi_4(s) = -\arcsin(\frac{sd_1}{R}),\ \xi_5(s) = -\arcsin(\frac{sd_2}{Rc_1}) \\ &\xi_6(s) = -\arcsin(\frac{sd_3}{Rc_1c_2}), \end{aligned} \tag{6}$$

where $c_1 = \sqrt{1-(sd_1/R)^2}, c_2 = \sqrt{1-(sd_2/(Rc_1))^2}$.

As a result, planning is carried out over the 4-dim space $\bar{\mathcal{X}} \subset \mathbb{R}^4$ with $\bar{X} = (x, y, \theta_0, s)^\top \in \bar{\mathcal{X}}$, i.e., the tree has 4-dim nodes. Note that the trailer system state still evolves in $\mathcal{X} \subset \mathbb{R}^6$ during the transition between nodes.

B. Motion Primitives

An MP can be viewed as a function $mp(\cdot)$ that transforms a node $\bar{X}_i$ to a new node $\bar{X}_j$, i.e., $\bar{X}_j = mp(\bar{X}_i)$. To ensure all nodes in i-AGT remains in the 4-dim space $\bar{\mathcal{X}}$, one should design $\mathcal{M}$ so that that $\bar{\mathcal{X}}$ is $\mathcal{M}$-invariant, while satisfying (2). That is $mp(\bar{X}) \in \bar{\mathcal{X}}, \forall \bar{X} \in \bar{\mathcal{X}}, mp \in \mathcal{M}$. Below illustrates how to obtain such $\mathcal{M}$ for on-lattice exploration and for off-lattice exploration, respectively, by solving a multitude of steering problems.

The Steering Problem. Steering refers to connecting two states with a kinematically or dynamically feasible trajectory. It can be posed as an open-loop optimal control problem (OCP). We denote the initial state as X_0, the target state as X_f, an admissible control set as $\mathcal{U}$, an objective function as $c(X, u)$, the control input as u, and final time as t_f. If the final time is free, the steering problem is cast into a fixed final time OCP via time-scale transformation $\gamma = \frac{t}{t_f}$. The time-scaled OCP has a final time 1:

$$\begin{aligned} \min_{u, t_f} \quad & c(X, u) \\ s.t. \quad & \frac{\mathrm{d}X}{\mathrm{d}\gamma} = t_f f(X, u)\ \&\ (3)\ \&\ (4) \\ & X(0) = X_0,\ X(1) = X_f,\ u \in \mathcal{U},\ t_f \in [0, \bar{t}_f], \end{aligned} \tag{7}$$

where $\bar{t}_f$ is a finite constant. The OCP (7) is formulated and solved by using CasADi [27] and IPOPT [28]. Notice that the MPs generated consider inputs u, allowing the solution from i-AGT to be directly executable.

When solving (7) numerically in ξ-coordinates, we choose 50 time steps over $[0, 1]$, and the bounds

$$\begin{aligned} &|x| \le 10,\ |y| \le 10,\ |\theta_0| \le 2\pi \\ &|\xi_i| \le \frac{\pi}{2}, \quad 4 \le i \le 6 \\ &|v| \le v_{\max} = 5R,\ |s| \le 1. \end{aligned} \tag{8}$$

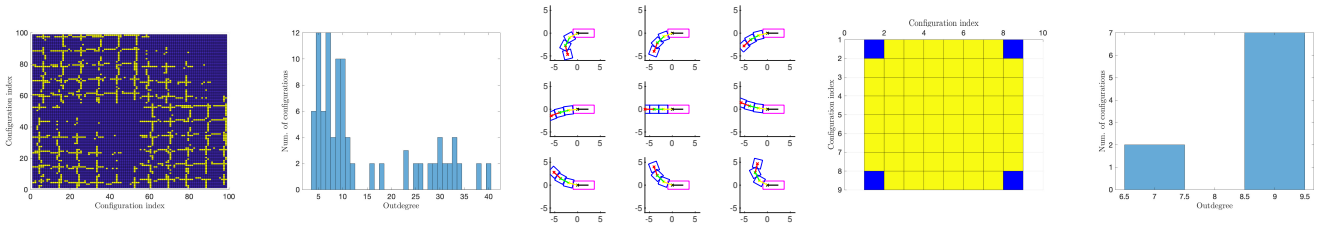
The key of constructing $\mathcal{M}$ is to determine *what* are the MPs and design the state-lattice accordingly, so that we can obtain the underlying MPs by solving the steering OCPs from one state-lattice node to the other.

State-lattice and Simplification. The main concern when designing the state-lattice is to ensure discrepancy and dispersion of the underlying MPs. Here we employ uniform discretization of a compact set: $\mathcal{D}_0 \triangleq [-L_x, L_x] \times [-L_y, L_y] \times [-\pi, \pi] \times [-1, 1] \subset \bar{\mathcal{X}}$. Denote the lattice set $\mathcal{S}$. Steering problems are defined with $(\bar{X}_0, \bar{X}_f) \in \mathcal{S} \times \mathcal{S}$.

The number of elements in $\mathcal{S}$ could be huge. For example, with $L_x = L_y = 2m$ and the resolution of the state-lattice $\Delta x = \Delta y = 1m, \Delta\theta = \pi/8$, and $\Delta s = 0.5$, we obtain a total of 2125 nodes. Solving steering problems by trying all possible combinations of $(\bar{X}_0, \bar{X}_f)$ will lead to solving millions of steering problems – computationally prohibitive. We simplify the MP generation process by exploiting the properties of the steering problems: invariance w.r.t. (x, y), symmetry against x-axis, $\pi/2$ rotation, and reversibility over time. In the following, we treat the x, y positions, the heading θ, and steering s as functions of time duration within the MP, i.e., $(x(t), y(t), \theta_0(t), s(t)),\ t \in [0, 1]$.

Solving steering problems from $\bar{X}_0 \in \mathcal{D}_2 \triangleq \{0\} \times \{0\} \times [0, \pi/2) \times [-1, 1] \to \bar{X}_f \in \mathcal{D}_0$ gives a group of MPs: $\mathcal{M}_{on}$. Similar to [5], [9], [22], one can categorize all MPs according to $\bar{X}_0$, i.e., each class of MPs is associated with a unique 2-dim pose $q = (\theta_0, s)$. During node expansion, one first performs $\mod(\theta_0, \frac{\pi}{2})$ to map $\bar{X}$ into $\mathcal{D}_2$, then retrieves the corresponding MPs as $\mathcal{M}_{on,q}$, and finally applies $\mathcal{M}_{on,q}$ at $\bar{X}$. In such a way, all nodes explored during planning remains on lattice.

Off-Lattice Motion Primitives. Applying MPs in an on-lattice manner is beneficial to maintaining the tree sparsity and high computational efficiency. However, restricting all nodes to lattice introduces significant limitations which compromised feasibility in tight environments and results in long paths with multiple cusps. One can remedy these shortcomings by adopting a large number of MPs for better



(a) (b) (c) (d) (e)

Fig. 2: Reachability analysis of $\mathcal{M}_{on}$ and $\mathcal{M}_{off}$: (a) adjacency matrix of $\mathcal{M}_{on}$ (b) outdegree of $\mathcal{M}_{on}$, (c) 9 poses of $\mathcal{M}_{off}$, (d) adjacency matrix of $\mathcal{M}_{off}$, and (e) outdegree of $\mathcal{M}_{off}$.

resolution completeness and potentially better path quality. However, ensuring the quality and connectivity of the MPs requires an even more complicated process.

To overcome such curse of dimensionality, the planner should plan over 4-dim space, not over the lattice within the 4-dim space. MPs should be classified and applied in a way that off-lattice nodes can be generated. This is readily achievable by grouping MPs according to steering s: given $\bar{X}$, one fetches and applies the MPs beginning with s at $\bar{X}$. In this case, each class of MPs should associate with a unique 1-dim pose $q = (s)$. We construct $\mathcal{M}_{off}$ from $\mathcal{M}_{on}$ by merging all MPs beginning with the same s to one class, and pruning MPs that ends close. In this work, we follow the above process to obtain $\mathcal{M}_{on}$ containing 2904 MPs and $\mathcal{M}_{off}$ containing 594 MPs. The analysis below shows that $\mathcal{M}_{off}$ contains enough MPs.

Remark 3.1: Generating $\mathcal{M}_{off}$ from $\mathcal{M}_{on}$ does not alleviate the computation complexity of MP generation. However, off-lattice application allows us to apply MPs that are generated with $\theta_0 = 0$ to other configuration with the same pose. Therefore, one can directly solve much less steering problems by further restricting $\bar{X}_0$ to the set: $\{0\} \times \{0\} \times \{0\} \times [-1, 1]$.

Remark 3.2: To keep the element in $\mathcal{M}_{off}$ simple enough to be followed by the underlying control system, we remove the MPs that contain cusps.

Analysis. Both $\mathcal{M}_{on}$ and $\mathcal{M}_{off}$ have to meet certain criteria for reasonable planning performance. We analyze them in the graph framework. Given an MP set $\mathcal{M}$, one can construct a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ where the node set $\mathcal{V}$ represents all possible poses q of $\bar{X}_0$, and $\mathcal{E}$ is a collection of directed edges from q_i to q_j where each of them represents the existence of an MP allowing the uni-directional transition from q_i to q_j.

The graph $\mathcal{G}$ constructed from $\mathcal{M}$ shall satisfy the following properties: 1) the graph is connected, meaning the trailer can transform from one pose to another pose in finite steps; 2) all nodes have similar in-degree and out-degree, meaning the transformation from pose to pose can potentially be done easily; and 3) all q have similar amount of backward and forward MPs, meaning the trailer can potentially move easily in 2D space. We construct $\mathcal{G}_{on}, \mathcal{G}_{off}$ from $\mathcal{M}_{on}, \mathcal{M}_{off}$, respectively. Fig. 2 shows the analysis results, when θ_0 and s are discretized over $[-\frac{\pi}{2}, \frac{\pi}{2}]$ and $[-1, 1]$ with resolution $\frac{\pi}{12}$ and $\frac{1}{4}$, respectively. As a result, $\mathcal{M}_{on}$ ends up with 99 2-dim poses q, and $\mathcal{M}_{off}$ contains 9 1-dim poses. Fig. 2(a) and (d) plot the adjacency matrices of $\mathcal{M}_{on}$ and $\mathcal{M}_{off}$, respectively where yellow in (i,j) means the transition from q_i to q_j exists. We can see that the yellow area in Fig. 2(d) covers almost matrix, meaning that without considering obstacles, the trailer can transform from one pose to another pose by applying at most two MPs. One easily confirms the connectivity of $\mathcal{M}_{on}, \mathcal{M}_{off}$ by checking the indegree and outdegree of all nodes. Fig. 2(b) and (d) show the outdegree of $\mathcal{M}_{on}$ and $\mathcal{M}_{off}$, respectively, where $\mathcal{M}_{on}$ exhibits much more non-uniformity than $\mathcal{M}_{off}$. Fig. 2(c) illustrates 9 poses used in $\mathcal{M}_{off}$. It is noteworthy that $\mathcal{M}_{off}$ is better in the sense that it is fully connected with much less (5x) MPs.

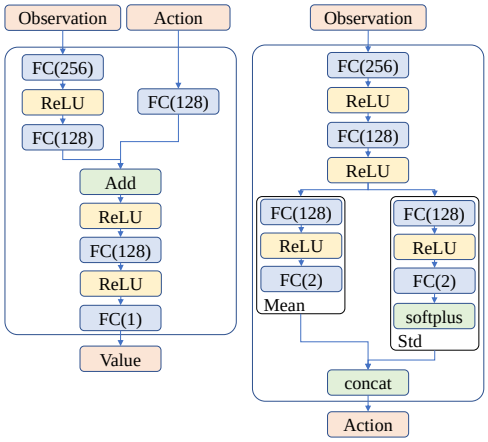
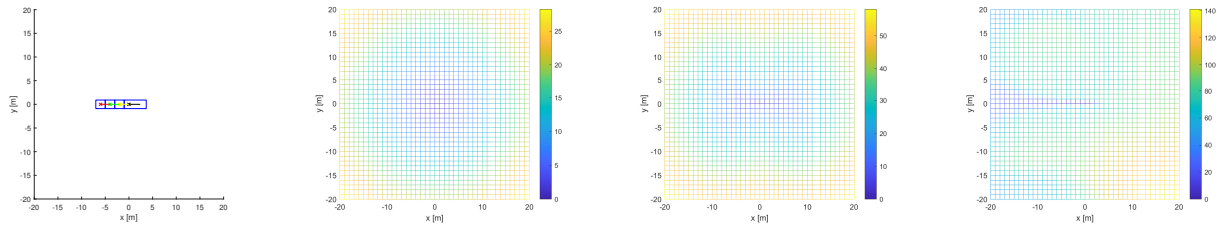


Fig. 3: The critic net (left) and actor net (right).

C. Mode and Estimated Cost-to-go

The planning efficiency of i-AGT is highly dependent on the mode definition/selection to find the right subset of MPs to apply; and the estimation accuracy of the cost-to-go to find the right node to apply these MPs. In this work, we applied the idea in [10] and classify the MPs into two modes: *forward mode* and *backward mode*. A child node generated following II-D is likely to explore the forward MPs if the parent node was also exploring forward, so that the planner won't easily "undo" its previous exploration. A more involved mode definition/selection can be developed, which may be future work. On the other hand, constructing a heuristic function to approximate the cost-to-go for trailers is much more challenging and time consuming than for cars, because the steering problem does not admit an analytical solution. We propose to learn the navigation policy and value function in free space with the soft actor-critic (SAC) algorithm [29], which is a model-free, online, off-policy,



(a) (b) (c) (d)

Fig. 4: Example of the heuristic value for goal configuration at $(0, 0, 0, 0, 0, 0)^\top$ based on: (b) Euclidean distance, (c) RS path, and (d) SAC value function.

actor-critic reinforcement learning method. Previous works have considered learning polices for similar systems with deep deterministic policy gradient (DDPG) [30] or deep Q-Network (DQN) [31]. Since our goal is to obtain a value function to approximate the cost-to-go, we favour exploration property and continuous action space, and hence choose SAC over DDPG or DQN.

To learn the navigation policy, we initialize the trailer state randomly in $\mathcal{X}$. The network structure is shown in Fig. 3, where observation refers to the state X and action refers to the input u. "FC(n)" refers to a fully connected layer with output size n. "Add" refers to an addition layer, which adds inputs from multiple neural network layers element-wise. The "softplus" layer applies the softplus activation function $y = \log(1 + e^x)$, which ensures the positiveness of outputs. The "concat" refers to the concatenation layer, which takes inputs and concatenates them along a specified dimension. Since the zero heading and steering configuration is often a preferred trailer parking pose, reward is given when the trailer is closed to $X_{train,goal} = (0, 0, 0, 0, 0, 0)^\top$. Each episode terminates when the maximum steps per episode $MaxSteps$ is reached, when the trailer goes out of a $35[m]$x$35[m]$ window centered at the goal, or when the trailer runs into a jack knife configuration. The design of the reward function and the training process are crucial to the performance of the trained policy. The two objectives of the reward function are:

- encouraging the trailer to go to $X_{train,goal}$ as quickly as possible;
- not discouraging exploration of the complicated trailer kinematics and the control policy.

We design a sparse reward function. While $MaxSteps$ is not reached, the reward at time k with observation X_k is:

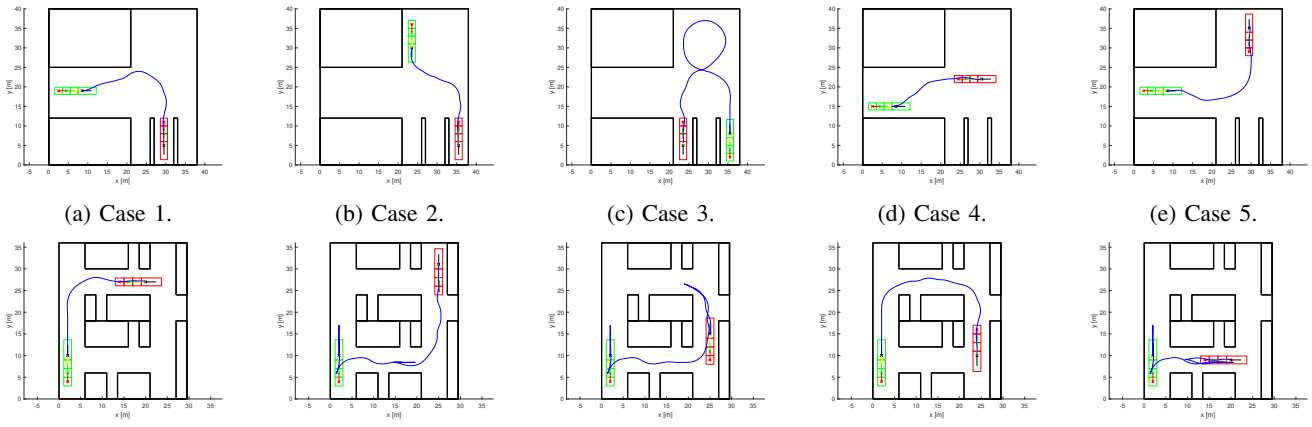
$$r(X_k) = \begin{cases} 1 & \text{if } \|X_k - X_{train,goal}\|_{W_1} \leq \epsilon \\ 0 & \text{otherwise} \end{cases}, \tag{9}$$

where $W_1 = \text{diag}([4.83, 4.25, 0.33, 0.15, 0.07, 0.03])$ is a weight matrix and ϵ determines the size of the "goal region." Since the reward is sparse, it is hard to receive reward at the beginning if ϵ is too small. Therefore, we start the training with $\epsilon = 1$ until the average reward converges, then shrink ϵ to half of its previous value, and repeat the process until $\epsilon = 0.25$.

To obtain the approximated cost-to-go from the learned critic value function, we use a zero vector for the action input and use the resulting critic value function as the heuristic value. We compare the proposed heuristic with heuristic functions based on Euclidean distance and Reeds-Shepp (RS) path [32], respectively. Fig. 4(b), (c), and (d) show the heuristic value to reach $X_{train,goal} = (0, 0, 0, 0, 0, 0)^\top$ starting from $\bar{X} = (x(0), y(0), 0, 0)$ as an example. The color indicates the heuristic value, where the locations with lower cost-to-go are colored in blue and higher cost-to-go are colored towards yellow. The heuristic value provided by Euclidean distance is shown in Fig. 4(b), where the value change is the same in all directions from the origin. This indicates that this heuristic cannot reflect the different levels of difficulty in maneuvering tasks when the trailer heading and steering angles are different. The heuristic value provided by RS path length is shown in Fig. 4(c), which has smaller values along the x-axis and larger values with location close to y-axis. This reflects the difficulty in terms of steering for a car model and similarly for the trailer. However, this heuristic cannot represent the steering angle, which largely affects the maneuvering difficulty. The heuristic based on the value function learned by SAC is shown in Fig. 4(d), which has lower values along the x-axis because the trailer only needs to drive straight to the goal. It also captures the fact that the trailer may need a larger space for maneuvers even though the Euclidean distance to the goal and the RS path length are small. This is reflected, for example, by $(x, y) = (-20, -5)$ having a smaller value than $(0, -5)$ in Fig. 4(d), which is not captured by the other two heuristics. Thus, the learned value function can more accurately reflect the true cost-to-goal. Note that the learned heuristic also has a higher value when the trailer needs to go backward. Although there is no preference between forward and backward motions during planning, in reality, forward motions are preferred because they are easier to execute.

IV. Empirical Evaluation

The proposed method is tested by simulation to benchmark the performance of the i-AGT with the proposed heuristic (i-AGT-NN) against the baseline i-AGTs that use RS as heuristic, ones with on-lattice (i-AGT-RS-on) MPs and the other with off-lattice (i-AGT-RS-off) MPs. The simulation environments are designed to mimic a tracker-trailer moving matirials in a large factory area where it needs to navigate through narrow aisles and park into narrow spaces. This section presents several of the simulation results as



(a) Case 1. (b) Case 2. (c) Case 3. (d) Case 4. (e) Case 5.

(f) Case 7. (g) Case 8. (f) Case 9. (h) Case 10. (i) Case 11.

Fig. 5: Simulation results of 10 cases. The initial position of the trailer is colored in green and the goal position in colored in red. The blue line indicates the trajectory of the tractor.

TABLE I: Comparison i-AGT performance with heuristic based on RS and SAC value function

Case No.	i-AGT-RS-on			i-AGT-RS-off			i-AGT-NN		
	Planning Time [s]	# of Nodes Explored	Path length [m]	Planning Time [s]	# of Nodes Explored	Path length [m]	Planning Time [s]	# of Nodes Explored	Path length [m]
1	N/A*	N/A	N/A	14.44	2094 (18475)	35.67	2.02	319 (972)	38.65
2	0.50	67 (410)	37.68	4.08	585 (6666)	29.37	0.63	79 (317)	33.03
3	N/A	N/A	N/A	274.22	17381 (514137)	71.62	130.42	9400 (265622)	78.00
4	2.44	274 (5714)	36.27	103.88	7625 (198423)	58.33	0.53	68 (335)	28.79
5	2.98	179 (4812)	90.27	324.78	19551 (623083)	68.58	2.028	300 (912)	47.32
6	2.80	166 (5722)	88.15	5.49	383 (8598)	40.93	8.31	581 (13588)	69.71
7	1.54	98 (2536)	57.19	5.57	413 (5770)	45.19	2.46	182 (2239)	45.19
8	N/A	N/A	N/A	49.67	2285 (93292)	71.03	14.71	1043 (19528)	75.00
9	N/A	N/A	N/A	N/A	N/A	N/A	66.52	2859 (134787)	79.05
10	2.89	2094 (4812)	90.27	2.49	172 (2531)	52.09	4.54	325 (4739)	52.09
11	N/A	N/A	N/A	6.78	466 (8285)	69.52	1.59	124 (1883)	61.87

* Solver failed or time out. The maximum computation time is 350 seconds.

examples. The tractor-trailer parameters used $L = 2.396[m]$ and $d_1 = d_2 = d_3 = 2[m]$. The MPs are processed following the proposed method, which leads us to an $\mathcal{M}_{on}$ containing 2904 MPs and an $\mathcal{M}_{off}$ containing 594 MPs. The algorithms terminate when any node on the tree X satisfies $\|X - X_f\|_2 \leq \epsilon$, where $\epsilon = 0.2$. Simulation is conducted on a 10-core Intel i9 3.7GHz desktop with Matlab R2021a. Fig. 5 shows the simulation results with i-AGT-NN in 10 cases, where the initial position of the trailer is colored in green and the goal position is in red. Table I shows the detailed planner performance.

We first compare i-AGT-NN and i-AGT-RS-off to verify the effectiveness of the proposed heuristic. Fig. 5(a)-(e) show the cases for trailer parking where several narrow parking spots are in the bottom right and an open space is on the top right allowing for maneuvering. Case 1 and 2 require the trailer to perform a right turn parking and a parallel parking, respectively. In both cases, i-AGT-NN outperforms i-AGT-RS-off in terms of planning time and requires less node exploration with similar resulting path length. Case 3 moves the trailer from one parking spot to another. In order to change the heading angle by 180 degrees, the trailer must utilize the empty space, which makes the planning problem very challenging, because moving trailer to the upper boundary for successful planning, is against the direction suggested by the heuristic of both heuristics. It takes both i-AGT-NN and i-AGT-RS-off a longer time to solve the problem comparing to other test cases. However, i-AGT-NN requires 50% less time than i-AGT-RS-off. Case 4 and 5 show the importance of having a heuristic that accurately captures the system kinematics. Both cases require similar navigation skills as case 2 and case 1, respectively. While i-AGT-NN performs similarly to cases 2 and 1, i-AGT-RS-off suffers a longer planning time and path length because the heuristic cannot capture the steering error. In previous cases 1 and 2, the narrow space around the goal forced the steering to be closed to zero. However, when the goal is in a relatively free space, the results of case 4 and 5 shows that it is important to have an accurate heuristic to efficiently guide the search. Fig. 5(f)-(i) show the cases of trailer navigating around narrow aisles. All planners perform similarly in cases 6, 7, 10, and 11, while the planning time is shown to be largely decreased by i-AGT-NN in case 8 and 9. The path length from both planners are also similar in cases $7\sim11$.

i-AGT-RS-on has short planning time in the successful cases, but it fails to find a solution in nearly half of the cases due to insufficient resolution, and results in long paths

in cases 5,7,10. Using the NN heuristic may help reducing path length but will not improve the success rate. Note that $\mathcal{M}_{off}$ is derived from $\mathcal{M}_{on}$, and these results show that allowing off-lattice exploration truly requires less MPs to reach the same level of (or even improve) planning success rate. In summary, i-AGT-NN outperforms i-AGT-RS-off in terms of average planning time while being able to keep the path length at the same level, and it outperforms i-AGT-RS-on in terms of success rate and path length. While the computation time is acceptable for off-line planning in static environments, the computation load is still heavy for reactive planning in dynamic environments.

V. Conclusion and Future Work

This paper presented a motion planning strategy that utilized improved A-Search Guided Tree to enable autonomous parking of a standard 3-trailer system with a car-like tractor. While exploiting the well-established lattice idea to circumvent the curse of dimensionality, we proposed to perform planning over 4-dim space instead of planning over 4-dim lattice by allowing the planner to explore outside of the lattice. This is achieved by constructing motion primitives dependent only on the steering angle. This drastically lowers the complexity in the generation and selection of motion primitives; leads to better success rate; and typically results in improved paths. To further increase search efficiency, we described a data-driven heuristic modeling the maneuver cost of the trailer to capture the cost-to-go by training a neural network through reinforcement learning. Simulations demonstrated the effectiveness of the proposed method in terms of success rate, planning speed, and path length.

References

[1] R. M. Murray and S. S. Sastry, "Nonholonomic motion planning: Steering using sinusoids," *IEEE transactions on Automatic Control*, vol. 38, no. 5, pp. 700–716, 1993.

[2] D. Tilbury, R. M. Murray, and S. S. Sastry, "Trajectory generation for the n-trailer problem using goursat normal form," *IEEE Transactions on Automatic Control*, vol. 40, no. 5, pp. 802–819, 1995.

[3] S. Sekhavat, P. Svestka, J.-P. Laumond, and M. H. Overmars, "Multilevel path planning for nonholonomic robots using semiholonomic subsystems," *The international journal of robotics research*, vol. 17, no. 8, pp. 840–857, 1998.

[4] F. Lamiraux, S. Sekhavat, and J.-P. Laumond, "Motion planning and control for hilare pulling a trailer," *IEEE Transactions on robotics and automation*, vol. 15, no. 4, pp. 640–652, 1999.

[5] M. Cirillo, T. Uras, and S. Koenig, "A lattice-based approach to multi-robot motion planning for non-holonomic vehicles," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 232–239.

[6] M. Cirillo, "From videogames to autonomous trucks: A new algorithm for lattice-based motion planning," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 148–153.

[7] O. Ljungqvist, N. Evestedt, M. Cirillo, D. Axehill, and O. Holmer, "Lattice-based motion planning for a general 2-trailer system," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 819–824.

[8] P. Töws and D. Zöbel, "Reversing general 2-trailer vehicles using a 2d steering function model and a novel mesh search algorithm," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021, pp. 1274–1281.

[9] O. Ljungqvist, N. Evestedt, D. Axehill, M. Cirillo, and H. Pettersson, "A path planning and path-following control framework for a general 2-trailer with a car-like tractor," *Journal of field robotics*, vol. 36, no. 8, pp. 1345–1377, 2019.

[10] Y. Wang, "Improved a-search guided tree construction for kinodynamic planning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5530–5536.

[11] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[12] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[13] B. Li, Y. Zhang, T. Acarma, Q. Kong, and Y. Zhang, "Trajectory planning for a tractor with multiple trailers in extremely narrow environments: A unified approach," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8557–8562.

[14] R. Oliveira, O. Ljungqvist, P. F. Lima, and B. Wahlberg, "Optimization-based on-road path planning for articulated vehicles," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 572–15 579, 2020.

[15] B. Li, L. Li, T. Acarman, Z. Shao, and M. Yue, "Optimization-based maneuver planning for a tractor-trailer vehicle in a curvy tunnel: A weak reliance on sampling and search," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 706–713, 2021.

[16] A. V. Rao, "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, vol. 135, no. 1, pp. 497–528, 2009.

[17] X. Zhang, A. Liniger, A. Sakai, and F. Borrelli, "Autonomous parking using optimization-based collision avoidance," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 4327–4332.

[18] J. Leu, G. Zhang, L. Sun, and M. Tomizuka, "Efficient robot motion planning via sampling and optimization," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 4196–4202.

[19] J.-W. Choi and K. Huhtala, "Constrained global path optimization for articulated steering vehicles," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 1868–1879, 2015.

[20] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[21] F. Islam, V. Narayanan, and M. Likhachev, "A*-connect: Bounded suboptimal bidirectional heuristic search," in *2016 IEEE International Conference On Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2752–2758.

[22] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *Journal of Field Robotics*, vol. 26, no. 3, pp. 308–333, 2009.

[23] K. Bergman, O. Ljungqvist, and D. Axehill, "Improved path planning by tightly combining lattice-based path planning and optimal control," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 1, pp. 57–66, 2020.

[24] P. Rouchon, M. Fliess, J. Lévine, and P. Martin, "Flatness and motion planning: the car with n trailers," in *Proc. ECC'93, Groningen*, 1993, pp. 1518–1522.

[25] C. Altafini, A. Speranzon, and B. Wahlberg, "A feedback control scheme for reversing a truck and trailer vehicle," *IEEE Transactions on robotics and automation*, vol. 17, no. 6, pp. 915–922, 2001.

[26] Z. Wang, A. Ahmad, R. Quirynen, Y. Wang, S. Di Cairano, A. Bhagat, E. Zeino, and Y. Zushi, "Motion planning and model predictive control for automated tractor-trailer hitching maneuve," *IEEE Conference on Control Technology and Applications (CCTA)*, August 2022.

[27] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[28] A. Wachter, "An interior point algorithm for large-scale nonlinear optimization with applications in process engineering," Ph.D. dissertation, Carnegie Mellon University, 2002.

[29] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[30] E. Bejar and A. Moran, "A preview neuro-fuzzy controller based on deep reinforcement learning for backing up a truck-trailer vehicle," in *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*. IEEE, 2019, pp. 1–4.

[31] C.-J. Hoel, K. Wolff, and L. Laine, "Automated speed and lane change decision making using deep reinforcement learning," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2148–2155.

[32] J. Reeds and L. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pacific journal of mathematics*, vol. 145, no. 2, pp. 367–393, 1990.