Mixed-Integer and Conditional Trajectory Planning for an Autonomous Mining Truck in Loading/Dumping Scenarios: A Global Optimization Approach

Bai Li, *Member, IEEE*, Yakun Ouyang, *Student Member, IEEE*, Xiaohui Li, Dongpu Cao, *Member, IEEE*, Tantan Zhang, and Yaonan Wang

***Abstract*—Trajectory planning for a heavy-duty mining truck near the loading/dumping sites of an open-pit mine is difficult. As opposed to trajectory planning for a small-sized passenger car in a parking lot, trajectory planning for a heavy-duty mining truck involves complex factors in vehicle kinematics and environment. These factors make the concerned trajectory planning scheme a mixed-integer nonlinear program (MINLP) incorporated with conditional constraints (denoted as C-MINLP). MINLP solvers can neither deal with conditional constraints nor find global optima in real time. Instead of solving the C-MINLP directly, we build a from-coarse-to-fine framework so that the coupled difficulties (the mixed integral variables, conditional constraints, and the demand for global optimality) are divided and conquered. At the coarse search stage, a global-optimality-enhanced hybrid A* search algorithm is proposed to find a near-optimal coarse trajectory with the mixed integral variables, conditional kinematic constraints, and global optimality considered. The coarse trajectory is further polished at the refinement stage, wherein the nominal C-MINLP is simplified as a small-scale NLP. The solution to the NLP is an optimized trajectory, which does not violate the complex constraints in the nominal C-MINLP. This indicates that conversion from the C-MINLP to an NLP is efficient with the help of a high-quality coarse trajectory.**

***Index Terms*—Trajectory planning, autonomous mining vehicle, computational optimal control, mixed-integer nonlinear program.**

Manuscript received 1 September 2022; revised 29 September 2022; accepted 12 October 2022. Date of publication 14 October 2022; date of current version 20 March 2023. This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 531118010509, in part by the National Natural Science Foundation of China under Grants 62103139 and 62003362, in part by the Natural Science Foundation of Hunan Province, China under Grant 2021JJ40114, and in part by the 2022 Opening Foundation of State Key Laboratory of Management and Control for Complex Systems under Grant E2S9021119. *(Corresponding author: Dongpu Cao.)*

Bai Li is with the State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Changsha 410082, China, and also with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China (e-mail: libai@zju.edu.cn).

Yakun Ouyang and Tantan Zhang are with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, China (e-mail: yakun@hnu.edu.cn; zhangtantan@hnu.edu.cn).

Xiaohui Li is with the College of Intelligence Science, National University of Defense Technology, Changsha 410073, China (e-mail: xiaohui_lee@outlook.com).

Dongpu Cao is with the School of Vehicle and Mobility, Tsinghua University, Beijing 100084, China (e-mail: dongpu.cao@uwaterloo.ca).

Yaonan Wang is with the College of Electrical and Information Engineering, Hunan University, Changsha 410082, China, and also with the National Engineering Laboratory for Robot Visual Perception and Control, Changsha 410082, China (e-mail: yaonan@hnu.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TIV.2022.3214777.

Digital Object Identifier 10.1109/TIV.2022.3214777

I. Introduction

MINING trucks are typically used in open-pit mines for loading, hauling, and dumping ores [1], [2], [3], [4], [5], [6]. Open-pit mines are generally harsh locations filled with dust, noise, and rockfalls, thus leading to risks and challenges for people who manually drive mining trucks. The deployment of autonomous mining trucks is promising to reduce the overall labor requirements and improve production efficiency [3], [7], [8].

Trajectory planning is about generating a spatio-temporal and continuous curve that is collision-free and kinematically feasible for a vehicle to reach its goal pose from the initial pose [9]. In an open-pit mine, trajectory planning toward the loading/dumping sire is challenging due to the complex constraints and demands. An autonomous truck generally needs multiple maneuvers before it reaches the goal configuration. Such tasks are quite different from on-road cruising and thus are commonly referred to as automated parking [10], [11], [12]. This study is about parking trajectory planning for a heavy-duty mining truck close to the loading/dumping site in an open-pit mine.

A. Motivations

Parking a heavy-duty mining truck differs from parking a small-sized passenger car due to the complex demands and constraints related to vehicular kinematics and environment.

1) Complexity in Vehicle Kinematics: A parking trajectory should be curvature-bounded to satisfy the kinematic principle of a front-steering vehicle [13]. The kinematic constraints of a heavy-duty truck include some additional rules. First, the distance between adjacent cusps along a planned trajectory should be sufficiently long, otherwise the trajectory cannot be perfectly tracked by a low-level controller due to the dead-zone effect of electrical motors. Second, the curvature bound is set

smaller when the vehicle moves backward than forward. This is because driving a heavy-duty vehicle backward deserves to be conservative [14].

2) Complexity in Cost Function: Basically, a parking trajectory should be optimal w.r.t. the traverse distance/time, energy, comfort, and safety. Besides that, trajectory planning for a heavy-duty truck considers the following few aspects. First, the number of maneuvers should be minimized. This is because each intermediate maneuver for a heavy-duty truck consumes time, energy, and wear of brake pads. Second, each backward trajectory segment is penalized if its length is longer than a specified threshold. This is because 1) backward tracking control is relatively difficult; 2) backward driving for long distances differs from human drivers' habits, which may cause troubles in human-machine interactions. Notably, the dislike of long backward driving segments is just a cost rather than a hard constraint.

3) Complexity in Environmental Constraints: The workspace of a mining truck is usually unstructured and tiny in a loading/dumping site. Specifically, the goal pose is usually set close to the barrier of the drivable area, i.e., in a tiny slot. This requires that the collision risks between the ego truck and the obstacles should be accurately modeled. The workspace is also cluttered by small surface stones, which render multiple homotopy classes. Ideally, a planned parking trajectory should have an optimal homotopy class, otherwise the traverse efficiency is degraded [15], [16].

4) Difficulties in Trajectory Planning: Nominally, a trajectory planning scheme is formulated as a mathematical programming (MP) problem [17], which is presented as a nonlinear program (NLP) in most cases [18], [19]. But the aforementioned analysis in this subsection indicates that the MP contains integer decision variables (the maneuver times) and conditional constraints with if-else expressions. Thus, the MP concerned in this study is not an NLP but a mixed-integer NLP (MINLP) containing conditional constraints. For simplicity, the concerned MINLP with conditional constraints is denoted as C-MINLP in the remaining of this article.

Gradient-based optimizers are typically used to deal with MPs such as NLPs or MINLPs. However, the situation becomes challenging for a C-MINLP because the conditional constraints it contains are non-differentiable, which renders that a gradient-based solver is inefficient at the non-differentiable points [20]. Therefore, it deserves to investigate how to handle a C-MINLP efficiently.

In dealing with the concerned C-MINLP problem, one expects to derive a solution with global optimality rather than local optimality. Therefore, it deserves to investigate how to enhance or guarantee the global optimality of a C-MINLP solution.

The concerned trajectory planning scheme is launched on-site, thus it deserves to investigate how the C-MINLP could be solved in real time.

As a summary, the difficulty in trajectory planning for a heavy-duty vehicle in unstructured loading/dumping scenarios lies in how to efficiently and quickly find a globally optimal trajectory that can fulfill the complicated conditional demands and constraints.

B. Contributions

This study tackles the difficulties raised in the preceding subsection, i.e., to quickly find a globally optimal trajectory that can fulfill the complicated demands and constraints. Instead of directly solving the nominal C-MINLP, we decouple the C-MINLP into two layers, wherein the upper layer coarsely identifies a neighborhood of the global optimum and the lower layer quickly finds the optimum. Such from-coarse-to-fine trajectory planners are commonly seen in the community of autonomous driving [21], but our unique contributions are focused on 1) how to assign the difficult issues mentioned in Section I.A to the two layers so that they are separately resolved, and 2) how to guarantee that the already conquered issues in the upper layer are well inherited to the lower layer. The concrete contributions are as follows:

1) A global-optimality-enhanced hybrid A* (GHA) search algorithm is proposed to deal with the complex constraints and demands in the upper layer. Compared with the conventional hybrid A* (HA) search algorithm that returns just a feasible solution, GHA greedily enhances the global optimality of the solution until the maximum allowable runtime is reached.
2) An optimization-based method is proposed in the lower layer, which runs fast and ensures that the already addressed issues in the upper layer are not violated. Concretely in modeling the optimization problem, the integer decision variables and the conditional constraints are safely discarded according to the output of the upper layer. Through this, we quickly derive a near-global-optimal solution to the nominal C-MINLP via solving a small-scale NLP.

C. Organization

In the rest of this article, Section II reviews the related works. Section III states the trajectory planning problem in the form of a C-MINLP. Section IV introduces our proposed trajectory planner. Simulation and real-world experimental results are reported and discussed in Section V, followed by Section VI, wherein conclusions are drawn.

II. Related Works

This section reviews the prevalent path/trajectory planners suitable for automated parking in an unstructured scenario. The existing planners are typically classified as search-, sampling-, and optimization-based methods.

Search-based planners abstract the state space of the trajectory planning problem as a graph of nodes and search for a connection from the initial node to the goal node. Dolgov et al. [22] proposed HA, where penalties are incorporated into the cost-to-come function to repel path primitives with zig-zag patterns or long backward distances. The penalties in HA cover part of the complex constraints and costs mentioned in Section I.A. However, HA does not support setting a smaller curvature bound for backward primitives. Besides that, HA only finds a feasible but not optimal path, which reduces the productiveness of a

truck especially when the workspace is highly cluttered and/or multiple maneuvers are needed. HA also suffers from the curse of dimensionality. Concretely, the graph resolution should be set high when the workspace is tiny, otherwise no solution can be found [23], [24]; but high resolution renders an increase in the node number, which makes the search process largely slowed down. Typically, other search-based methods also suffer from the curse of dimensionality. To conclude, search-based planners are generally imbalanced between time efficiency and quality.

Sampling-based planners generate trajectory/path primitives and select the ones that guide the vehicle to the goal pose. Sampling-based planners are further classified as deterministic methods such as state lattice planner [25] and stochastic methods such as rapidly-exploring random tree (RRT) series [26]. Deterministic samplers are inflexible to handle the kinematic and collision-avoidance constraints unless abundant primitives are predefined, which suffers from the curse of dimensionality. Stochastic samplers are flexible to generate primitives and even optimal in probability. However, the randomness in a stochastic sampler harms its robustness and reproducibility. Thus sampling-based planners are imperfect.

An optimization-based planner solves the nominal MP via a gradient-based optimizer. In most cases, the MP is an NLP or a quadratic program (QP) [27], [28]. However, most gradient-based MP solvers run slowly in dealing with conditional constraints and costs, i.e., the ones involving if-else conditions [20]. Some studies [29], [30], [31], [32], [33] converted conditional constraints and costs into differentiable forms. However, [29] or [30] assumes that a high-quality initial guess is available, which is not always true. Shi et al. [31] or Li et al. [32] involves nested optimization, thus runs slowly. Guo et al. [33] uses the *logsumexp* function to approximate the conditional constraints in an NLP that involve the *max* operation, but that method is limited to a rather small branch of planning problems with axially aligned spatial constraints. More importantly, approximating the conditional constraints via *logsumexp* is neither strictly accurate nor tractable in the scale. Besides the inefficacy to address conditional constraints, most gradient-based optimizers only find local optima, thus an optimization-based planner hardly finds high-quality solutions if it is used alone.

Mixing multiple types of planners has been well considered to reduce the limitation of each single method type, but few studies considered how the planners are tightly combined, that is, how the intermediate result derived by an upstream planner is efficiently inherited toward a downstream one [34].

As an overall conclusion of this section, the existing methods are not qualified for a heavy-duty mining truck to plan high-quality trajectories quickly under complex constraints and/or costs.

III. Problem Statement

This section formulates the concerned trajectory planning scheme as a C-MINLP, which comprises a cost function $J(\mathbf{c}_i, \mathbf{n}_i, \mathbf{p})$ and constraints $g(\mathbf{c}_i, \mathbf{n}_i, \mathbf{p}) \leq 0$:

$$\begin{aligned} &\min J\left(\mathbf{c}_i, \mathbf{n}_i, \mathbf{p}\right), \\ &\text{s.t.,} g\left(\mathbf{c}_i, \mathbf{n}_i, \mathbf{p}\right) \leq 0. \end{aligned} \tag{1}$$

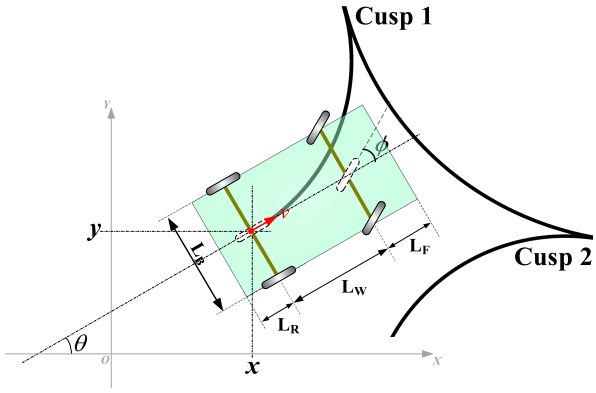


Fig. 1. Schematics on vehicle geometrics, kinematics, and the concept of cusp.

Herein, $\mathbf{c}_i \in \mathbb{R}^{\mathrm{n_c}}$ denotes the *i*th continuous variable, $\mathbf{n}_i \in \{0,1\}^{\mathrm{n_n}}$ represents the *i*th integer variable, and $\mathbf{p} \in \mathbb{R}^{\mathrm{n_p}}$ denotes continuous variables that are not related to *i*. A trajectory is characterized by a sequence of configurations $\mathbf{c}_i (i = 0, \ldots, \mathrm{N_{fe}})$. Concretely, $\mathbf{c}_i \equiv [x_i, y_i, \theta_i, v_i, a_i, \phi_i, \omega_i]$, wherein $(x_i,\ y_i)$ denotes the mid-point of the rear-wheel axle (Fig. 1), θ_i is the orientation angle, v_i is the vehicle velocity, a_i stands for the acceleration, ϕ_i denotes the steering angle, and ω_i is the angular velocity. $gs_i \in \mathbf{n}_i$ refers to a Boolean variable that shows whether a gear shift is happening. $T \in \mathbf{p}$ is a scalar that stands for the duration of the driving process. Solving the C-MINLP is about finding a set of $\mathbf{c}_i$, $\mathbf{n}_i$, and $\mathbf{p}$ that minimizes J while satisfying the constraints $g \leq 0$ strictly. The modeling details about the cost function and the constraints are presented in the rest of this section.

A. Kinematic Constraints

A single-track bicycle model is deployed to describe the basic kinematic rule of a truck. Here, the modeling accuracy of the single-track bicycle model is good because the skid-steering effect is slim when a truck moves slowly in a loading/dumping scenario [35]. Concretely, we have

$$\begin{aligned} x_{i+1} - x_i &= \Delta t \cdot v_i \cdot \cos\theta_i, \\ y_{i+1} - y_i &= \Delta t \cdot v_i \cdot \sin\theta_i, \\ \theta_{i+1} - \theta_i &= \Delta t \cdot v_i \cdot \frac{\tan\phi_i}{\mathrm{L_W}}, \\ v_{i+1} - v_i &= \Delta t \cdot a_i, \\ \phi_{i+1} - \phi_i &= \Delta t \cdot \omega_i,\ i = 0, \ldots, \mathrm{N_{fe}} - 1. \end{aligned} \tag{2}$$

In this equation, $\Delta t \equiv T/\mathrm{N_{fe}}$, $\mathrm{L_W}$ denotes the wheelbase. Other parameters related to the geometric size of the truck include the front hang length $\mathrm{L_F}$, rear hang length $\mathrm{L_R}$, and width $\mathrm{L_B}$.

Allowable intervals are imposed on $\mathbf{c}_i$:

$$\begin{aligned} \mathrm{v_{min}} \leq v_i &\leq \mathrm{v_{max}}, \\ |a_i| &\leq \mathrm{a_{max}}, \\ |\omega_i| &\leq \Omega_{\mathrm{max}}, \end{aligned} \tag{3a}$$

and

$$|\phi_i| \leq \begin{cases} \gamma \cdot \Phi_{\max}, \text{if} v_i \geq 0 \\ \Phi_{\max}, \text{else} \end{cases}. \tag{3b}$$

$\gamma \in (0,1)$ is a user-specified parameter that restricts the truck to drive with a smaller curvature when it goes backward [14], [36].

Besides (2) and (3), the distance between two adjacent cusps should be larger than a predefined threshold parameter $\mathrm{L_{ac}} > 0$. Cusps refer to the intermediate trajectory waypoints where the ego vehicle's velocity is instantaneously zero (Fig. 1). Due to the dead-zone effect of electric motors, a heavy-duty vehicle can hardly track too short trajectory segments; setting a lower bound on the traverse distance between adjacent cusps provides the tracking controller a longer distance and more time to reduce the dead-zone error. Let us define a cusp prior to modeling such constraints. Specifically, the Boolean gear-shift index gs_i is defined as

$$gs_i = \begin{cases} 1, \text{if} v_i \cdot v_{i-1} < 0 \\ 0, \text{else} \end{cases}. \tag{4}$$

A set *S* contains the indexes with $gs_i = 1$. Suppose that S_k and S_{k+1} are two adjacent indexes among all the $\mathrm{N_M}$ ones in *S*, then

$$\sum_{i=S_k}^{S_{k+1}} |v_i| \cdot \Delta t \geq \mathrm{L_{ac}}, k = 1, \ldots, \mathrm{N_M} - 1. \tag{5}$$

B. Two-Point Boundary Constraints

Equality constraints are imposed at $i = 0$ and $i = \mathrm{N_{fe}}$ to set the initial and goal configuration of the truck:

$$\begin{aligned} \mathbf{c}_0 &= [\mathrm{x_{init}}, \mathrm{y_{init}}, \theta_{\mathrm{init}}, 0, 0, 0, 0], \\ \mathbf{c}_{\mathrm{N_{fe}}} &= [\mathrm{x_{goal}}, \mathrm{y_{goal}}, \theta_{\mathrm{goal}}, 0, 0, 0, 0]. \end{aligned} \tag{6}$$

$[\mathrm{x_{init}}, \mathrm{y_{init}}, \theta_{\mathrm{init}}]$ refers to the initial pose of the truck and $[\mathrm{x_{goal}}, \mathrm{y_{goal}}, \theta_{\mathrm{goal}}]$ denotes the predefined goal pose at *T*.

C. Collision-Avoidance Constraints

Suppose that the workspace for parking the truck is denoted as $\Upsilon \in \mathbb{R}^2$, the space occupied by polygonal obstacles and non-drivable areas is $\Upsilon_{\mathrm{OBS}} \subset \Upsilon$, and $fp(\cdot) : \mathbb{R}^{\mathrm{n_z}} \to \mathbb{R}^2$ is a nonlinear mapping from ego vehicle configuration $\mathbf{c}_i \in \mathbb{R}^{\mathrm{n_c}}$ to its rectangular footprint then the nominal collision-avoidance constraint is

$$fp(\mathbf{c}_i) \subset \Upsilon \backslash \Upsilon_{\mathrm{OBS}}, i = 0, 1, .., \mathrm{N_{fe}}. \tag{7}$$

D. Cost Function

The cost function is a weighted sum of multiple polynomials. First, the driving process is expected to complete early, then we have

$$J_1 = T. \tag{8a}$$

Second, the driving motions should be comfortable and the planned trajectory should be smooth:

$$J_2 = \sum_{i=0}^{\mathrm{N_{fe}}} (a_i^2 + v_i^2 \cdot \omega_i^2). \tag{8b}$$

This term models driving comfort according to the ISO 2631-1 standard mentioned in [37].

Third, the number of maneuvers should be minimized. Let us define the maneuver number as

$$J_3 = \sum_{i=0}^{\mathrm{N_{fe}}} gs_i. \tag{8c}$$

Fourth, each backward segment is penalized if it is longer than a threshold $\mathrm{L_{bm}} > 0$. This cost term is used to encourage a human-like driving style:

$$J_4 = \sum_{i=2}^{\mathrm{N_M}} \left[\sum_{j=S_k}^{S_{k+1}} \lfloor -v_j \cdot \Delta t \rfloor - \mathrm{L_{bm}} \right]^2, \tag{8d}$$

wherein the operation $\lfloor \rfloor$ is defined as

$$\lfloor x \rfloor = \begin{cases} x, \text{if} x > 0 \\ 0, \text{else} \end{cases}. \tag{8e}$$

In (8d), the term $\sum_{j=S_k}^{S_{k+1}} \lfloor -v_j \cdot \Delta t \rfloor$ calculates the length of each backward segment. If the segment is backward, then the value of $\sum_{j=S_k}^{S_{k+1}} \lfloor -v_j \cdot \Delta t \rfloor$ is positive, otherwise the value is 0. J_4 penalizes the backward segments that are longer than $\mathrm{L_{bm}}$ quadratically.

Summarizing all the aforementioned four terms yields an overall cost function *J*:

$$J = \mathrm{w_1} \cdot J_1 + \mathrm{w_2} \cdot J_2 + \mathrm{w_3} \cdot J_3 + \mathrm{w_4} \cdot J_4, \tag{9}$$

where $\mathrm{w_1}, \mathrm{w_2}, \mathrm{w_3}, \mathrm{w_4} > 0$ are weighting parameters.

E. Overall Formulation

Combining the cost function (9) and the aforementioned constraints renders the following MP:

$$\begin{aligned} &\min_{\mathbf{c}_i, \mathbf{n}_i, \mathbf{p}} (9), \\ &\quad \text{s.t. Kinematic constraints } (2) - (5); \\ &\qquad \text{Boundary conditions } (6); \\ &\qquad \text{Collision - avoidance constraints } (7); \\ &\qquad \text{Internal relation constraints } (8). \end{aligned} \tag{10}$$

Herein, each gs_i is an integer variable; constraints (2) are nonlinear; constraints (3b), (4), or (8d) are conditional. Thus (10) is a typical C-MINLP.

IV. The Proposed Trajectory Planner

A. Overall Architecture and Novelty

Nominally, the formulated C-MINLP (10) should be solved by a gradient-based optimizer. But the prevalent MINLP optimizers are not designated to deal with conditional constraints, thus

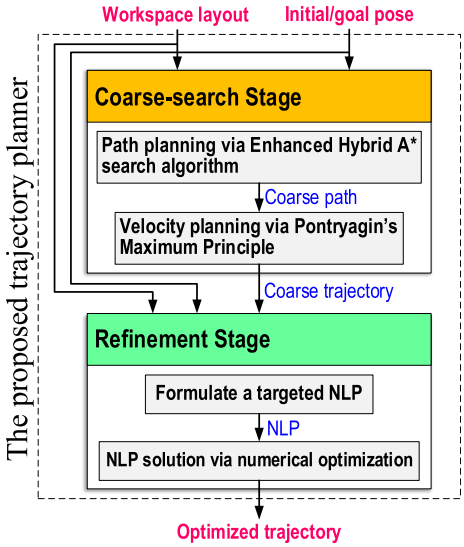


Fig. 2. Flowchart of the proposed trajectory planner.

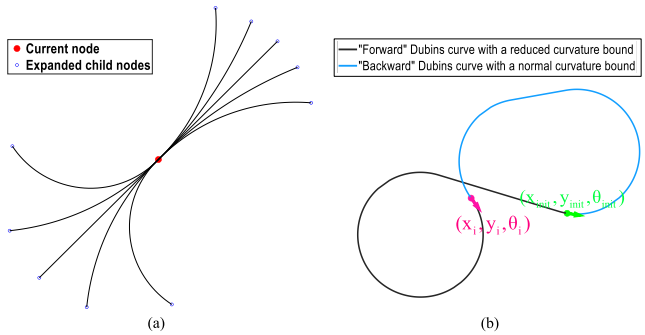


Fig. 3. Schematics on how GHA treats forward and backward path segments with different curvature bounds in (a) Node expansion procedure, and (b) Dubins connection procedure.

they run slowly or even fail. Besides that, the optimizers cannot guarantee to find the global optimum. Thus, solving the nominal C-MINLP directly is inapplicable.

This section proposes a two-stage trajectory planning method that aims to quickly find a high-quality solution to the concerned C-MINLP. The overall architecture is depicted in Fig. 2. A near-feasible trajectory close to the global optimum is derived at the coarse-search stage. Thereafter, the coarse trajectory is further polished at the refinement stage to achieve global optimality.

It deserves to emphasize that the two-stage architecture has been widely applied in the prevalent motion planners, thus the architecture itself is not novel. The core novelty lies in how to divide the complex and difficult elements in the C-MINLP into two parts so that they can be efficiently conquered at the two separate stages. Concretely, the coarse-search stage is responsible for getting a near-feasible trajectory. At that stage, the complex conditional constraints and cost function are considered; various domains of the solution space should be explored to enhance global optimality. After the coarse-search stage, the refinement stage optimizes the obtained coarse trajectory. Intuitively, the nominal C-MINLP should be solved at the refinement stage with the coarse trajectory serving as the initial guess. But solving a C-MINLP is time-consuming. Alternatively, we simplify the C-MINLP as a tractable NLP with the help of the coarse trajectory. The simplified NLP does not involve any integral decision variable or conditional constraints, thus it can be solved quickly to find a good solution to the nominal C-MINLP without constraint violations.

B. Coarse-Search Stage

The coarse-search stage generates a coarse trajectory by path-velocity decomposition [38]. That is, a coarse path is obtained by GHA and a time profile is attached to the path via Pontryagin's Maximum Principle (PMP) [39].

1) Path Planning Via GHA: GHA is a modified variant of HA. It is enhanced in the search ability to explore the global optimum. The complex constraints and cost terms in this work are also dealt with by GHA. GHA differs from HA in the five following aspects.

First, GHA swaps the initial and goal poses. This means that GHA searches from the goal pose toward the initial pose. Recall that the goal pose is usually close to obstacles, searching from such a tiny spot makes most of the early-iteration child nodes invalid because they collide with obstacles. The swap reduces the number of early-iteration nodes, thus making the search process more targeted in the early iterations.

Second, GHA fixes the cost-to-go function h to 0 constantly, i.e., $h \equiv 0$. Recall that the cost-to-go function h in HA is the maximum of the nonholonomic-without-obstacles cost and holonomic-with-obstacles cost [22], the calculation of which is time-consuming. Setting $h \equiv 0$ means that GHA does not estimate the cost-to-go cost at all, and thus GHA explores incrementally from the goal pose to the initial pose.

Third, GHA treats forward and backward motions differently. Given that forward path segments are allowed to have higher curvatures than backward ones according to (3b), the node expansion step generates forward and backward primitives with different curvature bounds (see Fig. 3(a)). Also, the Reeds-Shepp curve in HA is discarded because it cannot distinguish forward/backward curvature bounds. Instead of Reeds-Shepp curves, GHA deploys two Dubins curves with different curvatures. Fig. 3(b) depicts two Dubins curves that connect pose $(\mathrm{x_i}, \mathrm{y_i}, \theta_\mathrm{i})$ to the initial pose $(\mathrm{x_{init}}, \mathrm{y_{init}}, \theta_\mathrm{init})$. One Dubins curve connects $(\mathrm{x_i}, \mathrm{y_i}, \theta_\mathrm{i})$ to $(x_\mathrm{init}, y_\mathrm{init}, \theta_\mathrm{init})$ to represent a forward path while the other Dubins curve connects $(\mathrm{x_i}, \mathrm{y_i}, \theta_\mathrm{i} + \pi)$ to $(\mathrm{x_{init}}, \mathrm{y_{init}}, \theta_\mathrm{init} + \pi)$, thus representing a backward path. Note that the curvature of the forward path is smaller than that of the backward because we are exploring from the goal pose to the initial pose and "forward" paths are actually backward ones.

Fourth, the cost-to-come function g in GHA is modified. Recall that g in the conventional HA method penalizes frequent driving direction changes and frequent steering angle changes. In GHA, g additionally penalizes short distances between adjacent cusps and long backward path segments based on (5) and (8d).

Algorithm 1: Global-optimal Hybrid A* Search Algorithm.

Function $FeasiblePathSet \leftarrow$ GHA(map,[$x_{init}, y_{init}, \theta_{init}$], [$x_{goal}, y_{goal}, \theta_{goal}$])

Initialize $Openlist \leftarrow \oslash$, $FeasiblePathSet \leftarrow \oslash$, $iter \leftarrow 0$;
$node_0 \leftarrow$ RegisterNode($x_{init}, y_{init}, \theta_{init}$);
$Openlist \leftarrow Openlist \cup node_0$;
while ($FeasiblePathSet$.size() $< N_{fp}$) **do**
  $iter \leftarrow iter + 1$;
  $node_{\mathbf{cur}} \leftarrow$ ExtractCurBestNode($Openlist$);
  if ($iter\%20 == 0$) , **then** // Dubins curves are tried every 20 iterations
    $[path_1, path_2] \leftarrow$ ConnectDubinsCurves($node_{cur}, \Phi_{max}, \gamma$);
    if IsPathValid($path_1$, map) , **then**
      $complete\_path_1 \leftarrow$ FormCompletePath($path_1, node_{cur}$);
      $FeasiblePathSet \leftarrow FeasiblePathSet \cup complete\_path_1$;
    end if
    if IsPathValid($path_2$, map) , **then**
      $complete\_path_2 \leftarrow$ FormCompletePath($path_2, node_{cur}$);
      $FeasiblePathSet \leftarrow FeasiblePathSet \cup complete\_path_2$;
    end if
  end if
  $list_{forward} \leftarrow$ ExpandForwardChildNodes($node_{cur}, \Phi_{max}$);
  $list_{backward} \leftarrow$ ExpandBackwardChildNodes($node_{cur}, \Phi_{max} \cdot \gamma$);
  for each $node_{child} \in (list_{forward} \cup list_{backward})$, **do**
    if IsNodeClosed($node_{child}$), **then**
      continue;
    end if
    if IsNodeInOpenlist($node_{child}, Openlist$), **then**
      $[node_{child}, Openlist] \leftarrow$ TryChangeParent($node_{child}$);
    else if IsNodeValid($node_{child}$, map) , **then**
      $Openlist \leftarrow Openlist \cup node_{child}$;
    end if
  end for
  $Openlist \leftarrow Openlist \backslash node_{\mathbf{cur}}$;
  $node_{\mathbf{cur}}$.is_closed $\leftarrow 1$;
end while
return $FeasiblePathSet$.

Fifth, as opposed to the conventional HA that exits with a feasible path, GHA continues to search for other feasible paths after the first feasible one is derived.

The pseudo-codes of GHA are listed in Algorithm 1.

Remark 1: GHA can address all of the complex constraints and cost function terms mentioned in Section III. Concretely, GHA handles complex constraints by violation checks and penalties rather than gradient-based optimization, thus it is insensitive to the non-differentiability of the conditional constraints.

Remark 2: The usage of *FeasiblePathSet* in GHA brings chances to sort the one with a low cost among multiple feasible paths. GHA outputs with enhanced global optimality due to this strategy.

Remark 3: The paths that GHA outputs are said to be "feasible" in the sense of being collision-free, but each path is curvature-discontinuous, which is not really feasible in terms of vehicle kinematic constraint (2) [40].

2) Velocity Planning Via PMP: Each of the paths derived by GHA is attached to a time-optimal velocity profile via PMP [39] to form a candidate trajectory. The cost of each candidate trajectory is measured by (8). We greedily seek the trajectory with minimal cost value, which is the output of the coarse-search stage. For convenience, the output is denoted as $Traj_{\text{stage1}}$.

C. Refinement Stage

$Traj_{\text{stage1}}$ is curvature-discontinuous, thus it deserves to be polished at the refinement stage. An intuitive refinement way is to solve the C-MINLP (10) with $Traj_{\text{stage1}}$ serving as the initial guess. However, (10) contains conditional constraints, which are beyond the ability of a gradient-based MINLP solver. If one simply removes the conditional constraints from the C-MINLP before solving it, then the resultant trajectory may violate the conditional constraints. Hence, a challenge at the refinement stage is how to refine the coarse trajectory in a lightweight way while preserving the complex constraints already conquered at the coarse-search stage.

3) Targeted NLP Formulation: The refinement stage formulates a targeted NLP. Herein, the NLP is said "targeted" because it is targeted at preserving the satisfaction of the conditional constraints with the help of the coarse trajectory $Traj_{\text{stage1}}$.

In the first step, the coarse trajectory $Traj_{\text{stage1}}$ is resampled evenly along the time horizon into (N_{fe}+1) equidistant waypoints $\{wp_i | i = 0, \ldots, N_{fe}\}$. We require that each (x_i, y_i) stays close to the corresponding waypoint $\{wp_i\}$, that is,

$$\begin{aligned} wp_i.x - \beta \le x_i \le wp_i.x + \beta, \\ wp_i.y - \beta \le y_i \le wp_i.y + \beta, \\ i = 1, \ldots, N_{fe} - 1. \end{aligned} \tag{11}$$

$\beta > 0$ is a user-specified parameter that defines the size of the trust region. If one solves an NLP with the trust-region constraint (11), then the derived waypoints along the optimized trajectory stay close to those along $Traj_{\text{stage1}}$. Through this, constraints (5) and the cost function term (8d) that are already well considered at the coarse search stage need not be included in the targeted NLP formulation because (11) ensures that they are preserved.

In the second step, we request that the driving direction of the ego vehicle at the configuration points $\{v_i | i = 1, \ldots, N_{fe} - 1\}$ are consistent with those at the aforementioned waypoints

$\{\mathrm{wp}_i\}$, i.e.,

$$v_i \begin{cases} \geq 0, \text{if w } \mathrm{p}_i.v \geq 0 \\ < 0, \text{else} \end{cases},$$
$$i = 1, \ldots, \mathrm{N_{fe}} - 1. \tag{12}$$

Constraint (12) looks like an if-else expression, but it is actually not, because each $\mathrm{wp}_i.v$ is a known constant that determines the sign of each decision variable v_i. Imposing (12) in an NLP renders that the resultant maneuver times is the same as the one reflected in $Traj_{\mathrm{stage1}}$. Recall that large maneuver times have already been penalized by GHA, the integral variables gs_i and the corresponding cost function term (8c) can be safely removed from our targeted NLP formulation.

Since the sign of each v_i is fixed via (12), the conditional constraint (3b) is degraded as an ordinary linear constraint:

$$|\phi_i| \leq \begin{cases} \gamma \cdot \Phi_{\max}, \text{if w } \mathrm{p}_i.v \geq 0 \\ \Phi_{\max}, \text{else} \end{cases}. \tag{13}$$

The nominal collision-avoidance constraints (7) are modeled in a continuous and tractable way via safe travel corridors, the details of which are reported in [29].

Summarizing the aforementioned analysis renders the target NLP:

$$\begin{aligned} \min_{\mathbf{c}_i, \mathbf{n}_i, \mathbf{p}} \ & \mathrm{w}_1 \cdot J_1 + \mathrm{w}_2 \cdot J_2, \\ \text{s.t. } & \text{Kinematic constraints (2), (3a), (12), (13);} \\ & \text{Boundary conditions (6);} \\ & \text{Collision - avoidance constraints;} \\ & \text{Trust - region constraints (11).} \end{aligned} \tag{14}$$

4) Targeted NLP Solution: All the decision variables in the targeted NLP (13) are continuous. None of the constraints or cost function terms in (14) is conditional. Thus (14) is a regular NLP that can be easily dealt with by an NLP solver. This article adopts a well-known NLP solver, namely the interior-point method (IPM) [41], to solve the targeted NLP (13), during which $Traj_{\mathrm{stage1}}$ serves as the initial guess.

V. Experimental Results and Discussions

This section investigates the performance of the proposed trajectory planner in simulations and real-world experiments. Simulations are used to leverage the superiority of our method over existing ones while experimental results validate the closed-loop tracking performance. Typical results are reported at www.bilibili.com/video/BV13g411X7ho/.

A. Simulation Setup

A total of 20 simulation cases are generated, wherein the initial/goal pose and scenario layout are randomly defined.

Simulations were implemented in C++ and executed on an R7-5800H CPU that runs at 3.20 × 8 GHz. The NLP solver is an open-source software of IPM, namely IPOPT in version 3.13.4 [41], while the linear solver embedded in IPOPT is set to MA27 from Harwell Subroutine Library [42], which is called through a CasADi interface [43]. Typical parametric settings are listed in Table I.

TABLE I
Parametric Settings for Simulations

Parameter	Description and unit	Value
$\mathrm{L_F}$	Front hang length of the ego vehicle (m)	1.71
$\mathrm{L_W}$	Wheelbase of the ego vehicle (m)	5.73
$\mathrm{L_R}$	Rear hang length of the ego vehicle (m)	1.90
$\mathrm{L_B}$	Width of the ego vehicle (m)	3.50
$\mathrm{v_{min}}$, $\mathrm{v_{max}}$	Upper and lower bounds of v_i	-1.0, 2.0
$\mathrm{a_{max}}$	Upper bound of $\lvert a_i \rvert$ (m/s^2)	0.2
$\Phi_{\max}$	Upper bound of ϕ_i (rad)	0.49
$\Omega_{\max}$	Upper bound of $\lvert \omega_i \rvert$ (rad/s)	0.14
γ	Reduction multiplier in (3b)	0.6122
$\mathrm{L_{ac}}$	Minimal allowable distance between adjacent cusps (m)	5.0 m
$\mathrm{L_{bm}}$	Threshold distance for human-like backward path segment (m)	30.0 m
$\mathrm{w_1}$, $\mathrm{w_2}$, $\mathrm{w_3}$, $\mathrm{w_4}$	Weighting parameters in (9)	1, 0.025, 20, 1
$\mathrm{N_{fp}}$	Maximum feasible paths collected by GHA	200
$\mathrm{N_{fe}}$	Number of collocation points in building a C-MINLP (10)	100

TABLE II
Comparisons Between HA and GHA

	HA	GHA (N_{fp}=1)	GHA (N_{fp}=100)	GHA (N_{fp}=200)	GHA (N_{fp}=300)	GHA (N_{fp}=400)	GHA (N_{fp}=500)
Average CPU time (s)	0.013	0.117	0.216	0.276	0.331	0.382	0.437
Average cost	125.947	89.797	65.365	65.006	64.188	64.049	64.037

B. Comparative Simulations

The first round of simulations investigates the performance of GHA. We compare HA with GHA under various settings of $\mathrm{N_{fp}}$ and then list the results in Table II.

According to the results of HA and GHA under $\mathrm{N_{fp}} = 1$, GHA takes more time to find the first feasible path. This is because GHA encourages the node search process to grow from the goal pose incrementally via $h \equiv 0$. Although more runtime is spent, GHA returns with a lower average cost than HA. This indicates that the additional penalties imposed in GHA take effect. When $\mathrm{N_{fp}}$ is set larger, the cost value becomes smaller because more feasible solution candidates participate in the selection. Interestingly, it takes 117 ms to find the first feasible path while another 99 feasible paths are found in the next 99ms, which means that the average CPU time to find each feasible path is reduced. This phenomenon is reasonable. GHA search begins from the tiny goal spot, thus Dubins curves can hardly connect the initial pose. Thus the regular node expansion process is implemented in the first few iterations. When the best node in *Openlist* leaves the narrow goal spot and approaches the sparse workspace, Dubins connection trials become valid, thus feasible paths are derived more easily. Besides the aforementioned

TABLE III
COMPARISONS AMONG VARIANTS OF THE PROPOSED PLANNER

	Method A	Method B	Method C	This work
Success rate*	95%	95%	100%	100%
Average CPU time /s	0.278	1.246	0.615	0.571

* A solution process is regarded as failed if the adopted planner cannot find a feasible coarse path at the coarse-search stage or the NLP solver does not converge to feasibility at the refinement stage.

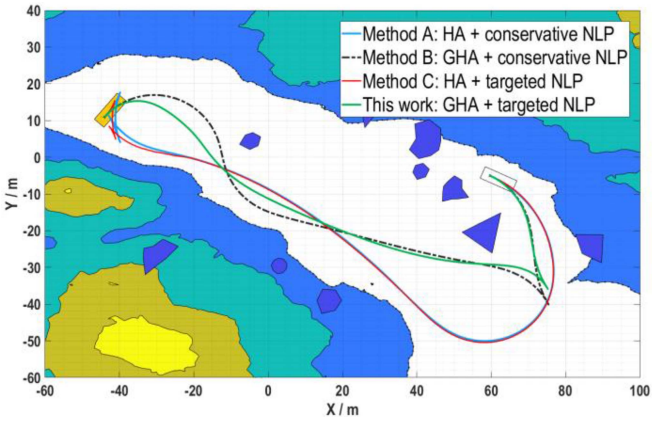


Fig. 4. Comparisons among the proposed planner and its variants. Methods A and C request redundant maneuvers due to the absence of a qualified coarse-search stage while method B cannot generate smooth trajectories because its refinement stage is too conservative.

analysis, Table II indicates that setting N_{fp} excessively large or small is not recommended while a medium value balances the solution speed and quality.

The second round of simulations investigates whether the two stages are tightly combined. Comparative methods are named A, B, and C, which are variants of our proposed method. Concretely, method A adopts HA (together with PMP) to find a coarse trajectory and adopts the conventional optimizer called the lightweight optimization method (LIOM) to refine it. Since LIOM cannot handle conditional constraint (3b), (3b) is replaced by a conservative constraint, which ensures that the output of LIMO never violates (3b):

$$|\phi_i| \leq \gamma \cdot \Phi_{\max}, i = 1, \ldots, N_{fe} - 1. \tag{15}$$

Method B is the same as method A, except that the coarse trajectory is derived by our proposed GHA rather than HA. Method C is the same as this work, except that the coarse trajectory is found by HA. The comparative simulation results are listed in Table III and are depicted in Fig. 4.

The comparison between method A and this work shows that 1) the coarse-search stage must provide a high-quality initial guess, otherwise the final output is still not good even after the refinement stage; and 2) simply reforming the conditional constraints as conservative ones is inefficient because the reformed constraint (15) is too strict, thus making the NLP infeasible.

As opposed to method A, method B improves the coarse-search stage, thus the refinement stage in method B receives a high-quality initial guess. However, method B runs slowly and suffers from NLP solution failures because the NLP problem

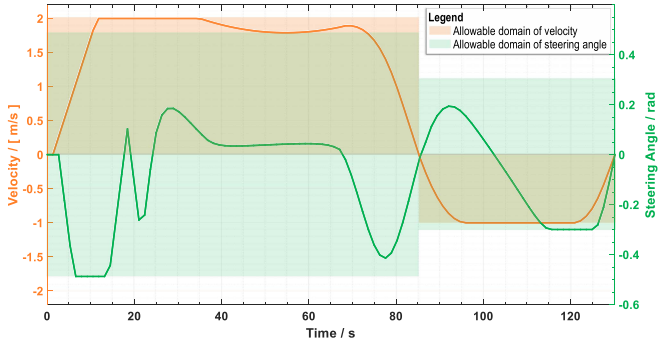


Fig. 5. Optimized steering angle and velocity profiles in association with the optimized trajectory. Notably the conditional constraint (3b) Is strictly satisfied.

solved at the refinement stage is too strict to make the NLP infeasible. A comparison between method B and this study indicates that tightly combing the two stages is really necessary in addition to having a high-quality initial guess.

Method C is strong in the refinement stage while weak in its coarse-search stage. Consequently, method C runs more slowly than our proposed planner with a lower solution quality as seen in Fig. 4. A comparison between method C and this study shows that enhancing the search quality at the coarse-search stage is sensible.

The aforementioned comparisons clearly show that our proposed planner is efficient in both stages and makes them compatible. Fig. 5 depicts the optimized steering angle profile together with the velocity, which indicates that our planner strictly satisfies the conditional constraint (3b). Two more typical simulation results are reported in Fig. 6.

C. Experimental Setup and Results

Experiments were conducted on a small-size autonomous vehicle platform in a $1.75\ \mathrm{m} \times 1.20\ \mathrm{m}$ indoor workspace (Fig. 7). Six infrared sensors from the NOKOV Motion Capture System were deployed for obstacle perception and ego vehicle localization. Concretely, reflective marking points were placed on the surface of each stone-like obstacle and the top of the autonomous vehicle (Fig. 8). The infrared light sources would emit infrared beams, which are captured by the infrared sensors after being reflected by the marking points. Through this, the location of each marking point is identified. The localization information was collected on a desktop PC, where trajectory planning was implemented. The trajectory planning module ran once to generate a trajectory before the ego vehicle began to move. The planned trajectory, together with the localization information would be sent from the PC to the ego vehicle platform via ZigBee communication technology. PID and pure pursuit controllers were used for longitudinal and lateral tracking under the frequency of 10 Hz, respectively.

Fig. 9 shows the closed-loop tracking performance. The dynamic tracking performance is demonstrated in the video link provided at the beginning of this section.

Experiments were also conducted in an open-pit mine in Jurong, China from October 2021 to September 2022. The

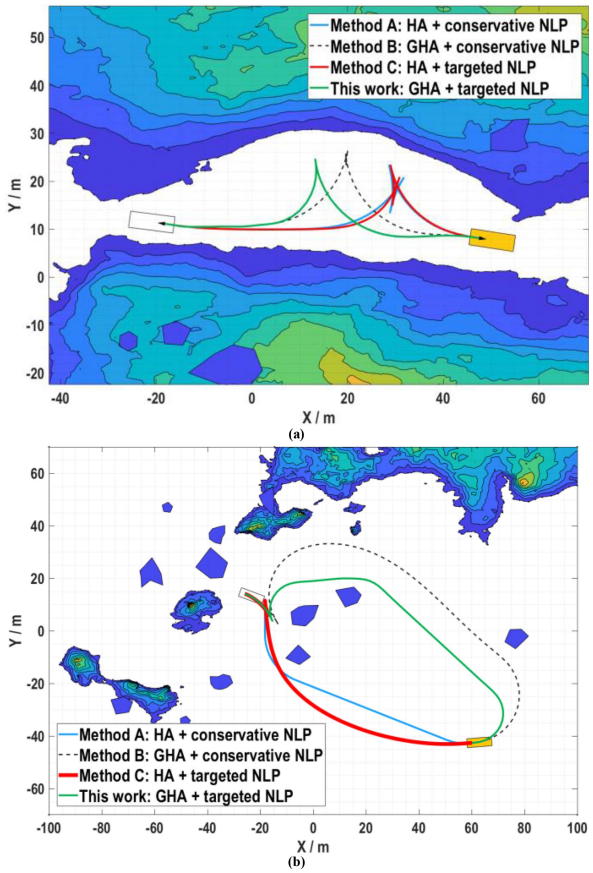


Fig. 6. Comparative simulation cases that how the differences among the proposed planner and its variants.

Fig. 7. An indoor workspace and a small-scale autonomous vehicle platform deployed for the field tests.

proposed trajectory planner was executed on an on-board Jetson AGX Xavier that runs at 2.27×8 GHz. The planner was executed once when the mining truck entered the loading or dumping region of the open-pit mine. Online trajectory replanning was implemented via parallel stitching [10] when the original trajectory became invalid due to drastic changes in the environment.

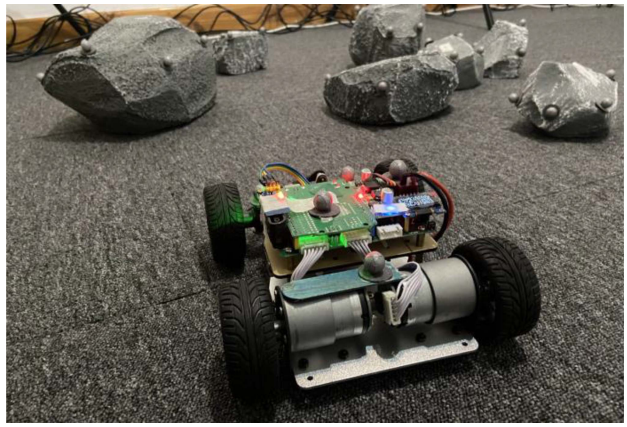


Fig. 8. Reflective marking points on the surface of each obstacle and on the top of the small-scale autonomous vehicle. Note that the marking points are deployed for ego vehicle localization and obstacle perception.

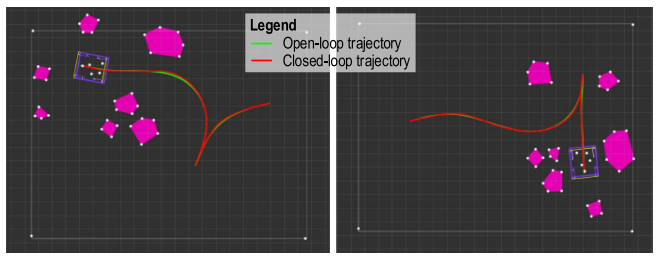


Fig. 9. Closed-loop tracking performances in the field tests (displayed in ROS Rviz monitor).

Empirically, relaxing the goal pose of the truck from a specific point to a small interval could enhance the real-time performance of the proposed planner at its refinement stage. Also, we notice that diversification-aware strategies are helpful to further enhance GHA in its search efficiency. Without such a strategy, the *FeasiblePathSet* in GHA would be quickly fulfilled by solutions that are highly similar to each other before diversified homotopy classes could be discovered.

VI. Conclusion

This article has proposed an on-site trajectory planner for a heavy-duty truck in loading/dumping schemes. The trajectory planning problem is nominally a C-MINLP, which must be quickly solved with global optimality. Instead of solving the C-MINLP directly, this work has managed to build a two-stage framework so that the coupled difficulties (the mixed integral variables, the conditional constraints, and the sensitivity to global optimality) are efficiently divided and conquered. Although there have been quite many planners that also deploy a two-stage architecture, they cannot handle the complex problems considered in this work, as we have proved via comparative simulations.

Acknowledgment

The authors are grateful to the anonymous reviewers and the associated editor for their valuable comments and suggestions.

Bai Li thanks Tianxing Yang, Yazhou Wang, Chen Li, Zhe Luo, Kun Li, and Xiaoyan Peng for their support in this study. Bai Li and Yakun Ouyang contribute equally to this article.

REFERENCES

[1] X. Zhang, A. Guo, Y. Ai, B. Tian, and L. Chen, "Real-time scheduling of autonomous mining trucks via flow allocation-accelerated Tabu search," *IEEE Trans. Intell. Veh.*, early access, Apr. 12, 2022, doi: 10.1109/TIV.2022.3166564.

[2] X. Ma, E. Huo, H. Yu, and H. Li, "Mining truck platooning patterns through massive trajectory data," *Knowl.-Based Syst.*, vol. 221, 2021, Art. no. 106972.

[3] P. F. Lima, M. Nilsson, M. Trincavelli, J. Mårtensson, and B. Wahlberg, "Spatial model predictive control for smooth and accurate steering of an autonomous truck," *IEEE Trans. Intell. Veh.*, vol. 2, no. 4, pp. 238–250, Dec. 2017.

[4] F. Tian et al., "Trajectory planning for autonomous mining trucks considering terrain constraints," *IEEE Trans. Intell. Veh.*, vol. 6, no. 4, pp. 772–786, Dec. 2021.

[5] D. Meng, Z. Pan, D. Cao, and L. Chen, "Berm detection for autonomous truck in surface mine dump area," in *Proc. IEEE Int. Intell. Transp. Syst. Conf.*, 2021, pp. 2829–2834.

[6] Y. Gao et al., "Parallel end-to-end autonomous mining: An IoT-oriented approach," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1011–1023, Feb. 2020.

[7] D. Bellamy and L. Pravica, "Assessing the impact of driverless haul trucks in Australian surface mining," *Resour. Policy*, vol. 36, no. 2, pp. 149–158, 2011.

[8] E. M. Nebot, "Surface mining: Main research issues for autonomous operations," in *Robot. Research*. Berlin, Germany: Springer, 2007, pp. 268–280.

[9] B. Yi, P. Bender, F. Bonarens, and C. Stiller, "Model predictive trajectory planning for automated driving," *IEEE Trans. Intell. Veh.*, vol. 4, no. 1, pp. 24–38, Mar. 2019.

[10] B. Li, Z. Yin, Y. Ouyang, Y. Zhang, X. Zhong, and S. Tang, "Online trajectory replanning for sudden environmental changes during automated parking: A parallel stitching method," *IEEE Trans. Intell. Veh.*, early access, Mar. 7, 2022, doi: 10.1109/TIV.2022.3156429.

[11] D. J. Kim, Y. W. Jeong, and C. C. Chung, "Lateral vehicle trajectory planning using a model predictive control scheme for an automated perpendicular parking system," *IEEE Trans. Ind. Electron.*, vol. 70, no. 2, pp. 1820–1829, Feb. 2023.

[12] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, and C. L. P. Chen, "Multiobjective optimal parking maneuver planning of autonomous wheeled vehicles," *IEEE Trans. Ind. Electron.*, vol. 67, no. 12, pp. 10809–10821, Dec. 2020.

[13] I. E. Paromtchik and C. Laugier, "Automatic parallel parking and returning to traffic manoeuvres," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 1997, vol. 3, pp. V21–V23.

[14] R. Rajamani, C. Zhu, and L. Alexander, "Lateral control of a backward driven front-steering vehicle," *Control Eng. Pract.*, vol. 11, no. 5, pp. 531–540, 2003.

[15] B. Li, Y. Ouyang, L. Li, and Y. Zhang, "Autonomous driving on curvy roads without reliance on Frenet frame: A cartesian-based trajectory planning method," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 15729–15741, Sep. 2022.

[16] K. Kolur, S. Chintalapudi, B. Boots, and M. Mukadam, "Online motion planning over multiple homotopy classes with Gaussian process inference," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2019, pp. 2358–2364.

[17] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[18] B. Li and Z. Shao, "Simultaneous dynamic optimization: A trajectory planning method for nonholonomic car-like robots," *Adv. Eng. Softw.*, vol. 87, pp. 30–42, 2015.

[19] X. Wang et al., "A symplectic pseudospectral method for nonlinear optimal control problems with inequality constraints," *ISA Trans.*, vol. 68, pp. 335–352, 2017.

[20] B. Li and Z. Shao, "A unified motion planning method for parking an autonomous vehicle in the presence of irregularly placed obstacles," *Knowl.-Based Syst.*, vol. 86, pp. 11–20, 2015.

[21] Y. Guo, D. D. Yao, B. Li, H. Gao, and L. Li, "Down-sized initialization for optimization- based unstructured trajectory planning by only optimizing critical variables," *IEEE Trans. Intell. Veh.*, early access, Mar. 29, 2022, doi: 10.1109/TIV.2022.3163335.

[22] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *Int. J. Robot. Res.*, vol. 29, no. 5, pp. 485–501, 2010.

[23] D. Le, Z. Liu, J. Jin, K. Zhang, and B. Zhang, "Historical improvement optimal motion planning with model predictive trajectory optimization for on-road autonomous vehicle," in *Proc. IEEE 45th Annu. Conf. Ind. Electron. Soc.*, 2019, pp. 5223–5230.

[24] Y. Lei, Y. Wang, S. Wu, X. Gu, and X. Qin, "A fuzzy logic-based adaptive dynamic window approach for path planning of automated driving mining truck," in *Proc. IEEE Int. Conf. Mechatronics*, 2021, pp. 1–6.

[25] M. Cirillo, "From video-games to autonomous trucks: A new algorithm for lattice-based motion planning," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 148–153.

[26] L. Han, Q. H. Do, and S. Mita, "Unified path planner for parking an autonomous vehicle based on RRT," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 5622–5627.

[27] B. Li et al., "On-road trajectory planning with spatio-temporal RRT* and always-feasible quadratic program," in *Proc. IEEE 16th Int. Conf. Automat. Sci. Eng.*, 2020, pp. 942–947.

[28] K. Kondak and G. Hommel, "Computation of time-optimal movements for autonomous parking of non-holonomic mobile platforms," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2001, vol. 3, pp. 2698–2703.

[29] B. Li et al., "Optimization-based trajectory planning for autonomous parking with irregularly placed obstacles: A lightweight iterative framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 11970–11981, Aug. 2022.

[30] X. Zhang, A. Liniger, A. Sakai, and F. Borrelli, "Autonomous parking using optimization-based collision avoidance," in *Proc. IEEE Conf. Decis. Control*, 2018, pp. 4327–4332.

[31] S. Shi, Y. Xiong, J. Chen, and C. Xiong, "A bilevel optimal motion planning (BOMP) model with application to autonomous parking," *Int. J. Intell. Robot. Appl.*, vol. 3, pp. 370–382, 2019.

[32] B. Li, K. Wang, and Z. Shao, "Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3263–3274, Nov. 2016.

[33] Y. Guo, D. Yao, B. Li, Z. He, H. Gao, and L. Li, "Trajectory planning for an autonomous vehicle in spatially constrained environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18326–18336, Oct. 2022.

[34] K. Bergman, O. Ljungqvist, and D. Axehill, "Improved path planning by tightly combining lattice-based path planning and optimal control," *IEEE Trans. Intell. Veh.*, vol. 6, no. 1, pp. 57–66, Mar. 2021.

[35] P. Polack, F. Altché, B. d'Andréa-Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 812–818.

[36] M. Kim, S. Shin, and J. Park, "Study on vehicle lateral control for backward driving," in *Proc. IEEE 13th Int. Conf. Ubiquitous Robot. Ambient Intell.*, 2016, pp. 191–193.

[37] B. Sakcak, L. Bascetta, G. Ferretti, and M. Prandini, "An admissible heuristic to improve convergence in kinodynamic planners using motion primitives," *IEEE Control Syst. Lett.*, vol. 4, no. 1, pp. 175–180, Jan. 2020.

[38] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *Int. J. Robot. Res.*, vol. 5, no. 3, pp. 72–89, 1986.

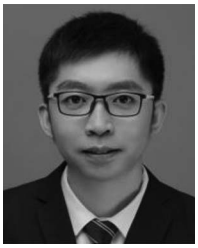
[39] L. S. Pontryagin, *Mathematical Theory of Optimal Processes*. Milton Park, U.K.: Taylor & Francis, 1987.

[40] G. Song and N. M. Amato, "Randomized motion planning for car-like robots with C-PRM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2001, vol. 1, pp. 37–42.

[41] A. Wächter and L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.

[42] HSL, "A collection of fortran codes for large scale scientific computation," 2007. [Online]. Available: http://www.hsl.rl.ac.uk/

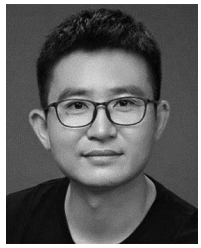
[43] J. Andersson et al., "CasADi: A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, 2019.



Bai Li (Member, IEEE) received the B.S. degree from the School of Advanced Engineering, Beihang University, Beijing, China, in 2013, and the Ph.D. degree from the College of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2018. From November 2016 to June 2017, he visited the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI, USA, as a joint training Ph.D. student. He is currently an Associate Professor with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha, China. Before teaching with Hunan University, he was with the JDX R&D Center of Automated Driving, JD Inc., China, as an Algorithm Engineer, from 2018 to 2020. He is the first author of more than 60 journal/conference papers and two books related to *numerical optimization*, *motion planning*, and *robotics*. His research focuses on numerical optimal control-based motion planning methods in autonomous driving. He was the recipient of the International Federation of Automatic Control (IFAC) 2014–2016 Best Journal Paper Prize from the Engineering Applications of Artificial Intelligence. He is currently an Associate Editor for IEEE TRANSACTIONS ON INTELLIGENT VEHICLES.

Yakun Ouyang (Student Member, IEEE) received the B.S. degree from the School of Information Engineering, Nanchang University, Nanchang, China, in 2020. He is currently working toward the master's degree with the College of Mechanical and Vehicle Engineering, Hunan University, Changsha, China. His research interests include decision making, trajectory planning, control, and software engineering of an autonomous vehicle system. He was the first-prize recipient of the 2019 National University Students Intelligent Car Race.

Xiaohui Li received the B.S. degree in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2009, and the Ph.D. degree in control science and engineering from the College of Mechatronics and Automation, Nation University of Defense Technology (NUDT), Changsha, China, in 2016. He has been a Visiting Ph.D. student with Ohio State University, Columbus, OH, USA. He is currently an Associate Professor with the College of Intelligence Science and Technology, NUDT. His research interests include the design and applications of complex control systems, decision-making, motion planning, and optimal control for autonomous vehicles. He was a Reviewer of a lot of Journals, including IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.



Dongpu Cao (Member, IEEE) received the Ph.D. degree from Concordia University, Montreal, QC, Canada, in 2008. He is currently a Professor with Tsinghua University, Beijing, China. His research interests include driver cognition, automated driving, and cognitive autonomous driving. He has contributed more than 200 papers and three books. He was the recipient of the SAE Arch T. Colwell Merit Award in 2012, IEEE VTS 2020 Best Vehicular Electronics Paper Award, and six Best Paper Awards from international conferences. Prof. Cao was the Deputy Editor-in-Chief for *IET Intelligent Transport Systems Journal*, and an Associate Editor for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE/ASME TRANSACTIONS ON MECHATRONICS, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE/CAA JOURNAL OF AUTOMATICA SINICA, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, and *ASME Journal of Dynamic Systems, Measurement, and Control*. Prof. Cao is also an IEEE VTS Distinguished Lecturer.

Tantan Zhang received the B.S. degree from Hunan University, Changsha, China, in 2012, the double M.S. degrees from the Politecnico di Torino, Turin, Italy, and Tongji University, Shanghai, China, in 2015, and the Ph.D. degree from the Politecnico di Torino, in 2020. He is currently an Assistant Professor with the College of Mechanical and Vehicle Engineering, Hunan University. His research focuses on motion planning of automated vehicles.

Yaonan Wang received the B.S. degree in computer engineering from the East China University of Science and Technology, Fuzhou, China, in 1981, and the M.S. and Ph.D. degrees in electrical engineering from Hunan University, Changsha, China, in 1990 and 1994, respectively. Since 1995, he has been a Professor with Hunan University. His research interests include robotics, intelligent perception and control, and computer vision for industrial applications. Since 2019, he has been an academician of the Chinese Academy of Engineering.