2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Planning Optimal Trajectories for Mobile Manipulators under End-effector Trajectory Continuity Constraint

Quang-Nam Nguyen[1] and Quang-Cuong Pham[1,2]

***Abstract*— Mobile manipulators have been employed in many applications that are traditionally performed by either multiple fixed-base robots or a large robotic system. This capability is enabled by the mobility of the mobile base. However, the mobile base also brings redundancy to the system, which makes mobile manipulator motion planning more challenging. In this paper, we tackle the mobile manipulator motion planning problem under the end-effector trajectory continuity constraint in which the end-effector is required to traverse a continuous task-space trajectory (time-parametrized path), such as in mobile printing or spraying applications. Our method decouples the problem into: (1) planning an optimal base trajectory subject to geometric task constraints, end-effector trajectory continuity constraint, collision avoidance, and base velocity constraint; which ensures that (2) a manipulator trajectory is computed subsequently based on the obtained base trajectory. To validate our method, we propose a discrete optimal base trajectory planning algorithm to solve several mobile printing tasks in hardware experiment and simulations.**

I. Introduction

A mobile manipulator often consists of one (or multiple) manipulator, such as a robotic arm, mounted on a mobile base which helps extend the workspace of the manipulator. This mobility enables mobile manipulators to be employed in many large-scale applications which are usually performed by multiple robots or a larger system, such as in printing [1]–[5], pick and place [6], [7], drilling [8], etc. However, the mobile base adds redundant degrees of freedom (DOFs) to the system, which makes the dimension of the configuration space higher than that of the task space. On one hand, this redundancy brings challenges to task and motion planning, but on the other hand, it leaves room for optimization [9].

In general, a robot accomplishes a high-level task (e.g. create multiple holes on a workpiece) by performing one or a sequence of sub-tasks (e.g. drill a hole, move to the next target). The high-level task is solved by finding the feasible or optimal sequence to perform the sub-tasks, which can be planned separately (task sequencing before motion planning) or in combined task and motion planning [10].

Due to the localization uncertainty of the mobile base (typically centimetres), it is often desirable to optimize the base motion based on an objective such as travel distance, control effort, number of base stops, etc. Optimization can be done in task sequencing and/or motion planning [11]. Optimal mobile manipulator task sequencing has been addressed in many applications such as pick and place, drilling, scanning [6]–[8]. For optimization in motion planning, an example is mobile printing (Fig. 1) in which the main task is to move the nozzle along a printing trajectory. In this case, we desire a slow and stable base motion while printing.

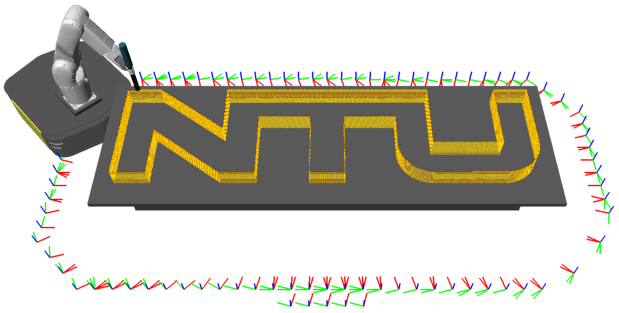


Fig. 1. Mobile 3D printing of NTU shape (10 layers, size: $3 \times 0.75 \times 0.15m$, total printing path length: $112.9m$) with constant nozzle speed of $10cm/s$.

In this paper, we focus on motion planning for motions that involve the *end-effector trajectory continuity constraint* which requires the end-effector to traverse a continuous trajectory (time-parametrized path) such as in mobile printing [1]–[3], [5] or spraying [12]. Note that this end-effector trajectory-continuous motion is different from the end-effector path-continuous motion in which the end-effector is only required to traverse a continuous path without specified velocity or timing information [13]–[17]. There are two approaches in mobile manipulator motion planning: whole-body planning and decoupled planning. These differences will be discussed in the next section.

Our contribution is an optimal base trajectory planning method for mobile manipulator motion planning under end-effector trajectory continuity constraint, which consists in:

- Using the manipulator's reachability to confine the base configurations inside an admissible region in configuration spacetime that satisfies end-effector trajectory continuity constraint, geometric task constraints, and collision avoidance; which ensures a manipulator trajectory can be planned subsequently.
- Proposing a discrete optimal base trajectory planning algorithm to solve the above problem with considerations for the base velocity constraint and optimization.

The remainder of this paper is organized as follows. In section II, we discuss related works. Section III formulates the problem and gives an overview of our method. In section IV, we propose a discrete optimal base trajectory planning algorithm. Section V validates our method in several mobile printing tasks and provides some evaluations.

[1]Singapore Centre for 3D Printing (SC3DP), Nanyang Technological University (NTU), Singapore nam.ngquang@gmail.com
[2]Eureka Robotics, Singapore

II. Related Works

A recent survey on mobile manipulator motion planning can be found in [18]. In some applications, short-term whole-body planning can be performed online during control [19]–[21]. In other applications that require long-term plans, high dimensionality becomes a challenge even for offline planning. To tackle this, an effective approach is to decouple the whole-body problem into base's and manipulator's motion planning problems which are, however, not entirely separated since their trajectories must ensure manipulator's reachability (under joint limits). This relationship can be represented by reachability map [22], inverse reachability map [23], or other geometric approximations [8], [16].

In our work, we use the geometric reachable region method [8] which is approximately equivalent to reachability and inverse reachability maps but has some advantages. Firstly, representing reachability by a geometric shape with geometric parameters (see Fig. 3) is light-weight compared to storing a reachability or inverse reachability map and matching them with the floor. Secondly, Inverse Kinematics (IK) solutions exist everywhere inside the geometric reachable region with 100% certainty, although in return for that, the end-effector's orientation must be within a specified range. A limitation of all above-mentioned methods is that they do not guarantee a continuous joint trajectory between IK solutions [24], which needs further research in the future.

In end-effector path-continuous tasks, the end-effector is required to track a given path. Using the decoupled approach, the base motion planning can be treated as a path planning problem and solved using variations of Rapidly-exploring Random Tree (RRT) planners [14], [15]. In the whole-body approach, the whole-body motion can be short-term planned and controlled online using Quadratic Programming (QP) [17], Model Predictive Control (MPC) [19], [20], or Sequential Linear Quadratic Optimal Control (SLQ) [21].

For end-effector trajectory-continuous tasks, the difference from path-continuous tasks is the time-parametrization along end-effector's trajectory. A major application of this class of tasks is mobile printing/spraying. Initial works in this field were performed with the base trajectory being planned manually [3], [12]. More recently in [1], [2], the authors followed the decoupled approach and used a two-step base motion planning method. Their method consists of a base path planning step using a variation of RRT* planner and a post-processing step that smooths and time-parametrizes the obtained path. However, the base kinematic constraints (e.g. velocity limits) were not considered during path planning which may results in unsuccessful post-processing. On the other hand, in the whole-body approach, while there are methods to compute pathwise-IK solutions for redundant systems [25], [26], extending these methods to mobile manipulation is not trivial since it will also face the challenge in handling kinematic constraints and optimization for the base. This direction can be investigated in future works.

In this paper, we would like to improve the decouple approach to address the above-mentioned limitations by formulating the base motion planning problem as a constrained optimal trajectory planning problem which allows us to consider optimization and multiple constraints (end-effector trajectory continuity constraint, geometric task constraints, collision avoidance, and base velocity constraint).

III. Methodology Overview

A. Configuration spacetime

The *configuration space (C-space)* $\mathcal{C}$ of a system is an n-dimensional manifold in which each point has n *generalized coordinates* representing the system configuration:

$$\mathbf{q} \coloneqq (q_1, ..., q_n) \in \mathcal{C} \tag{1}$$

Spacetime is a representation of the evolution of a dynamical system through time. The term *configuration spacetime (C-spacetime)* was first used in 1939 [27] and has been defined as the topological product of a real time axis $\mathcal{T} \subseteq \mathbb{R}$ and the configuration space: $\mathcal{X} \coloneqq \mathcal{T} \otimes \mathcal{C}$. Thus, the configuration spacetime is an $(n+1)$-manifold:

$$\mathcal{X} = \{\mathbf{x} \coloneqq (t, \mathbf{q}) \mid t \in \mathcal{T}, \mathbf{q} \in \mathcal{C}\} \tag{2}$$

where each point $\mathbf{x} = (t, q_1, ..., q_n)$ is called an *event* which occurs at time t (*temporal coordinate*) when the system is having the configuration $q_1, ..., q_n$ (*spatial coordinates*).

The *trajectory* of a system is a continuous sequence of events, which is a curve $\mathbf{q}(t)$, or equivalently, $\mathbf{x}(s)$ in C-spacetime. Here, the path parameter s goes from $s = 0$ at a start event to $s = 1$ at an end event (see Fig. 2). The tangent vector at any point along the trajectory:

$$\mathbf{x}' \coloneqq \frac{d\mathbf{x}}{ds} = \begin{pmatrix} t' \\ \mathbf{q}' \end{pmatrix} \tag{3}$$

must satisfy $t' \coloneqq dt/ds > 0$ since time is monotonic.

The *spacetime velocity* at any point along the trajectory is parallel to the tangent vector at that point:

$$\dot{\mathbf{x}} \coloneqq \frac{d\mathbf{x}}{dt} = \frac{\mathbf{x}'}{t'} = \begin{pmatrix} 1 \\ \dot{\mathbf{q}} \end{pmatrix} \tag{4}$$

where $\dot{\mathbf{q}} \coloneqq d\mathbf{q}/dt$ is called the *generalized velocity*.

B. Mobile manipulators in configuration spacetime

For example, our mobile manipulator consists of a 6-DOF manipulator $\mathbf{q}_m = (\theta_1, ..., \theta_6)$ and a 3-DOF planar mobile base $\mathbf{q}_b = (x, y, \varphi)$ where $\varphi \in \mathbb{S}^1$ is the orientation (yaw angle) of the base. Therefore, the whole-body configuration and C-space of a mobile manipulator are:

$$\mathbf{q} = (\theta_1, ..., \theta_6, x, y, \varphi) \in \mathcal{C}, \quad \mathcal{C} \subseteq \mathbb{R}^8 \otimes \mathbb{S}^1 \tag{5}$$

We follow the decoupled approach which treats the manipulator and the mobile base separately, i.e. $\mathcal{C} = \mathcal{M} \otimes \mathcal{B}$ where the *manipulator configuration space* is $\mathcal{M} \subseteq \mathbb{R}^6$ and the *base configuration space (B-space)* is $\mathcal{B} \subseteq \mathbb{R}^2 \otimes \mathbb{S}^1$.

We define the *base configuration spacetime (B-spacetime)*:

$$\mathbf{x} = (t, x, y, \varphi) \in \mathcal{X}, \quad \mathcal{X} \subseteq \mathbb{R}^3 \otimes \mathbb{S}^1 \tag{6}$$

where we denote $\mathbf{x}, \mathcal{X}$ instead of $\mathbf{x}_b, \mathcal{X}_b$ since we only need to apply the concept of spacetime on the mobile base.

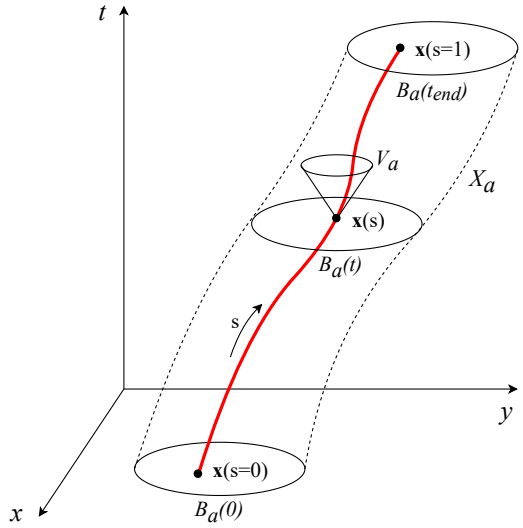


Fig. 2. Visualization of a trajectory in the base configuration spacetime (B-spacetime). The base orientation φ axis is not shown. Any point $\mathbf{x}(s)$ in the trajectory must be inside an admissible B-space $\mathcal{B}_a(t)$ and its velocity $\dot{\mathbf{x}}(s)$ must be inside a cone of admissible spacetime velocities $\mathcal{V}_a$. The admissible B-spacetime $\mathcal{X}_a$ is obtained as the admissible B-space changes in time.

C. Constraints

1) Geometric task constraints: In a 6D task, the end-effector follows as rigid body motions, so the task space is generally $SE(3)$. For 5D tasks, the geometric task constraints are end-effector's 3D position and vector orientation, so the task space is $\mathbb{R}^3 \otimes \mathbb{S}^2$. For example, printing is a 5D task in which the rotation around nozzle's axis is free. In this paper, we consider 5D tasks which bring more redundant DOFs to the problem ($n-5$ compared to $n-6$). Moreover, we notice that in printing, the nozzle's axis usually points vertically downwards which is parallel to the base's axis or only tilts slightly from it. This makes mobile printing tasks more challenging because the base can orient towards each end-effector pose at any angle: $\varphi \in \mathbb{S}^1$.

2) Collision avoidance: The base configuration spacetime allows us to consider collisions between the base and static or known dynamic obstacles by setting a safe distance from them. From static obstacles, we get an initial set of collision-free base configurations: $\mathcal{B}_{free}(0) \subseteq \mathcal{B}$. From known dynamic obstacles (e.g. printed structure), the sets of collision-free base configurations are obtained at different time instances:

$$\mathcal{B}_{free}(t) \subseteq \mathcal{B}_{free}(0),\ t \in \mathcal{T} \tag{7}$$

3) End-effector trajectory continuity constraint: The end-effector is required to follow a continuous task-space trajectory $\mathbf{p}(t)$ instead of just a geometric path. For each end-effector pose in its task-space trajectory $\mathbf{p}(t)$, the base configuration $\mathbf{q}_b(t)$ must be kept within a set of admissible configurations $\mathcal{B}_a(t) \subseteq \mathcal{B}_{free}(t)$ so that the end-effector can reach the pose without violating joint limits. The set of admissible events is therefore a subset of the B-spacetime which we call the *admissible B-spacetime*: (see Fig. 2)

$$\mathcal{X}_a = \{\mathbf{x} = (t, \mathbf{q}_b) \mid t \in \mathcal{T},\ \mathbf{q}_b \in \mathcal{B}_a(t)\} \subseteq \mathcal{X} \tag{8}$$

The admissible B-spacetime will be obtained based on the reachability analysis which will be explained in section VI.

4) Base velocity constraint: We consider the translational and rotational velocity limits of the mobile base which can be represented by a *set of admissible spacetime velocities*:

$$\mathcal{V}_a = \left\{\dot{\mathbf{x}} := (1, \dot{\mathbf{q}}_b) \mid \dot{x}^2 + \dot{y}^2 \le v_{max}^2,\ \dot{\varphi}^2 \le \omega_{max}^2\right\} \tag{9}$$

This constraint is visualized in Fig. 2: at every point on the trajectory, the spacetime velocity must be kept inside a cone.

D. Problem formulation

The goal of mobile manipulator motion planning is to find the whole-body joint trajectory so that the robot traverses the end-effector task-space trajectory subject to constraints and optimization. We decouple this problem into manipulator's and base's trajectory planning problems as follows.

The base trajectory planning step is solved first, during which we use the manipulator's kinematic reachability and other constraints to confine the base configurations inside an admissible B-spacetime so that the manipulator can reach the desired pose at any given time without violating joint limits.

1) Base trajectory planning: The *feasible trajectory planning* problem for mobile base is to find a base trajectory inside the admissible B-spacetime, such that the velocity at every trajectory point is an admissible velocity:

$$\mathbf{x}(s) \in \mathcal{X}_a, \quad \dot{\mathbf{x}}(s) \in \mathcal{V}_a \quad \forall s \in [0,1] \tag{10}$$

Handling constraints: geometric task constraints, end-effector trajectory continuity constraint, and collision avoidance are considered in $\mathcal{X}_a$; while base velocity constraint is realized by $\mathcal{V}_a$.

Optimal trajectory planning: among feasible solutions, it is desirable to find an optimal trajectory to minimize a cost:

$$\mathbf{x}^{opt}(s) = \arg\min_{\mathbf{x}(s)} J[\mathbf{x}(s)] \tag{11}$$

where $J[\mathbf{x}(s)]$ is the cost functional, typically formulated as an integral of a Lagrangian.

We use the cost functional for minimum control effort which is defined as: (we denote $a \cdot B \cdot a := a^T B a$)

$$\begin{aligned} J[\mathbf{q}_b(t)] &:= \int_0^{t_{end}} L(\mathbf{q}_b, \dot{\mathbf{q}}_b, t)dt = \int_0^{t_{end}} \dot{\mathbf{q}}_b \cdot \mathbf{I}_q \cdot \dot{\mathbf{q}}_b\, dt \\ \Leftrightarrow \quad J[\mathbf{x}(s)] &= \int_0^1 \mathbf{x}' \cdot \mathbf{I}_x \cdot \mathbf{x}' \frac{ds}{t'}, \quad \mathbf{I}_x = \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{I}_q \end{pmatrix} \end{aligned} \tag{12}$$

where the weight matrix $\mathbf{I}_q := diag(1,1,w)$ sets the relative weight between rotational and translational base motions. This cost functional for minimum control effort effectively prevents unnecessarily large velocity changes, since a slow and stable velocity profile results in a low control effort.

2) Manipulator trajectory planning: Given the obtained base trajectory, the manipulator trajectory planning problem is to find the joint trajectory of the manipulator so that its end-effector follows the desired task-space trajectory.

The method of using manipulator's reachability to confine the admissible base configurations guarantees that IK solutions exist for every pair of base and end-effector poses $\mathbf{q}_b(t), \mathbf{p}(t)$. Thus, the joint trajectory can be computed using fast IK solvers such as OpenRAVE's IKFast [28].

E. Solution approach

We notice that the problem formulation in this section returns a constrained optimal base trajectory planning problem. Since time always marches forward and the mobile base is confined inside the admissible B-spacetime which guides the base from start to goal, the difficulty does not lie on exploring a region to find the path to goal but on constraints and optimization. Therefore, we follow the discrete optimal planning approach [11] and propose a discrete optimal base trajectory planning algorithm based on dynamic programming.

IV. Optimal Base Trajectory Planning

A. Discretization

We sample $N+1$ points along the desired end-effector trajectory with uniform time-step size $\Delta t = t_{end}/N$. Thus, the corresponding discrete base trajectory must also consist of $N+1$ points (time steps): $i = 0, \ldots, N$. At time step i, the time is $t^i = i\Delta t$ and the path parameter is $s^i = i/N$.

We discretize $\mathcal{V}_a$ by firstly specifying a set of velocity step sizes $\Delta v_x, \Delta v_y, \Delta\omega$ then finding combinations of integers (k_x, k_y, k_φ) such that the corresponding admissible spacetime velocities $\dot{\mathbf{x}} := (1, k_x\Delta v_x, k_y\Delta v_y, k_\varphi\Delta\omega)$ satisfy the base velocity constraint in (9). Then, the set of admissible controls is:

$$\mathcal{U}_a = \{\mathbf{u} := \dot{\mathbf{x}}\Delta t \mid \dot{\mathbf{x}} \in \mathcal{V}_a\} \tag{13}$$

We set the step sizes of the spatial coordinates at:

$$\Delta x = \Delta v_x \Delta t, \quad \Delta y = \Delta v_y \Delta t, \quad \Delta\varphi = \Delta\omega\Delta t \tag{14}$$

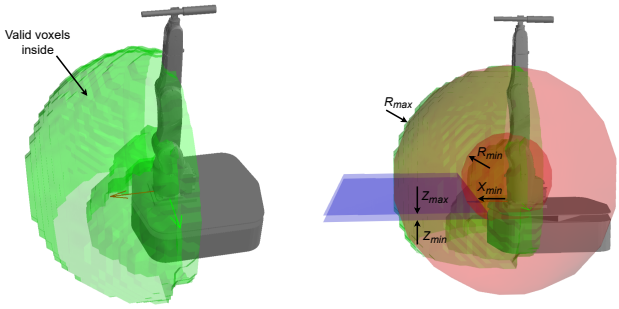
so that for any admissible controls, the mobile base moves from one grid point to another, i.e.

$$\mathbf{x}^{i+1} - \mathbf{x}^i = \dot{\mathbf{x}}^i \Delta t = \mathbf{u}^i \tag{15}$$

B. Kinematic Reachability Analysis

We implement the kinematic reachability analysis introduced in [8] as follows. Firstly, we discretize the space relative to the manipulator into 3D voxels with size $\delta \times \delta \times \delta$. Secondly, we specify a range of orientations based on the end-effector trajectory (geometric task constraints). For example, in printing, the nozzle mainly points downwards, and we allow small deviation from this main direction. Next, we compute Inverse Kinematics and mark the valid voxels at which the end-effector can reach within the range of orientations determined previously. At the end of this process, we obtain a valid voxel cloud (Fig. 3a) which can be stored and re-used for similar tasks with the same robot.

To use an obtained valid voxel cloud for a given task, we determine the *geometric reachable region* according to each discretized point in the end-effector trajectory (a desired end-effector pose). The geometric reachable region is obtained by slicing the valid voxels cloud using: 2 horizontal planes bounding the height h of the desired pose: $z = h \pm \delta/2$, 1 vertical plane to keep a safe distance X_{min} from the robot, and 2 spherical surfaces centred at the manipulator's second joint, with radii R_{min}, R_{max} to maximize the volume of the in-between region containing only valid voxels (see Fig. 3b).



(a) Valid voxels cloud (b) Geometric reachable region

Fig. 3. Visualization of Kinematic Reachability Analysis

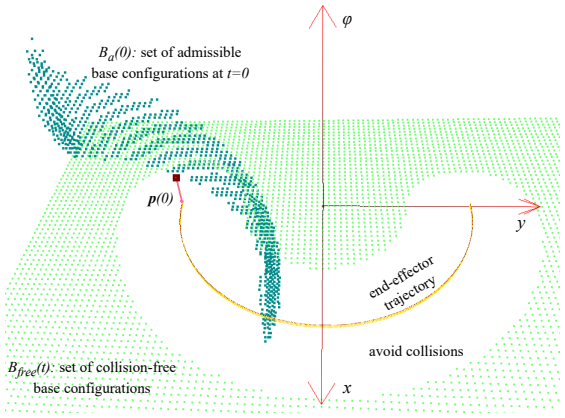


Fig. 4. Discretized admissible B-space $\mathcal{B}_a(0)$ for end-effector pose $\mathbf{p}(0)$. The vertical axis shows $\varphi \in (-\pi, \pi]$ for $\varphi \in \mathbb{S}^1$ while time axis is not shown.

Next, we obtain the discretized admissible B-space $\mathcal{B}_a^i$ (see Fig. 4) for each desired end-effector pose $\mathbf{p}(i\Delta t)$ as follows. Using the geometric reachable region parameters calculated for $\mathbf{p}(i\Delta t)$, we find all the collision-free base configurations where if the base is placed at, the desired end-effector pose is inside the geometric reachable region: $\mathbf{q}_b \in \mathcal{B}_a^i \subseteq \mathcal{B}_{free}(i\Delta t)$. Finally, the discretized admissible B-spacetime $\mathcal{X}_a$ is:

$$\mathcal{X}_a = \bigcup_{i=0}^{N} \mathcal{X}_a^i, \quad \mathcal{X}_a^i = \{(i\Delta t, \mathbf{q}_b) \mid \mathbf{q}_b \in \mathcal{B}_a^i\} \tag{16}$$

This step can be computed in negligible time in our tests.

C. Planning optimal mobile base trajectory

The numerical problem is: find the minimum-cost trajectory in a multi-source ($\mathbf{x}^0 \in \mathcal{X}_a^0$) multi-goal ($\mathbf{x}^N \in \mathcal{X}_a^N$) multi-stage ($i = 0, \ldots, N$) graph. The cost functional (12) becomes:

$$J = \sum_{i=0}^{N-1} l(\mathbf{u}^i) = \sum_{i=0}^{N-1} \mathbf{u}^i \cdot \mathbf{I}_x \cdot \mathbf{u}^i \frac{1}{\Delta t} \tag{17}$$

First, we introduce the following definitions which are inspired by similar concepts in control and planning [29].

Definition 1 (One-step feasible set): The *one-step feasible set* $\mathcal{Q}(\mathcal{I})$ is a set of admissible events $\mathbf{x} \in \mathcal{X}_a$ that can reach at least one event $\bar{\mathbf{x}} \in \mathcal{I}$ using one admissible control $\mathbf{u} \in \mathcal{U}_a$, that is $\bar{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) = \mathbf{x} + \mathbf{u}$ (*event transition equation*)

$$\mathcal{Q}(\mathcal{I}) = \{\mathbf{x} \mid f(\mathbf{x}, \mathbf{u}) \in \mathcal{I},\ \mathbf{x} \in \mathcal{X}_a,\ \mathbf{u} \in \mathcal{U}_a\} \tag{18}$$

Definition 2 (i-stage feasible set): The *i-stage feasible set* $\mathcal{K}^i(\mathcal{I}^N)$ is the set of admissible events that can reach at least one event in a final set $\mathcal{I}^N$ after a sequence of $N-i$ admissible controls. This set can be computed iteratively by

$$\begin{aligned}\mathcal{K}^N(\mathcal{I}^N) &= \mathcal{I}^N \cap \mathcal{X}_a \\ \mathcal{K}^i(\mathcal{I}^N) &= \mathcal{Q}(\mathcal{K}^{i+1}(\mathcal{I}^N))\end{aligned} \tag{19}$$

In our case, the goal set is $\mathcal{I}^N = \mathcal{X}_a^N$ so (19) becomes

$$\begin{aligned}\mathcal{K}^N(\mathcal{X}_a^N) &= \mathcal{X}_a^N \cap \mathcal{X}_a = \mathcal{X}_a^N \\ \mathcal{K}^i(\mathcal{X}_a^N) &= \mathcal{Q}(\mathcal{K}^{i+1}(\mathcal{X}_a^N)) \subseteq \mathcal{X}_a^i\end{aligned} \tag{20}$$

which suggests a multistage graph where its stages are the same as the time steps of the trajectory $i = 0, 1, ..., N$, and every node in the graph lies on a grid point of the discretized admissible B-spacetime ($\mathcal{K}^i \subseteq \mathcal{X}_a^i$).

Instead of constructing the graph before a subsequent graph search, we propose running *backward value iterations* as follows: while connecting the nodes using (20), concurrently calculate the *minimum cost-to-go* $G(\mathbf{x})$ [11] and store the *optimal next-event* $\bar{\mathbf{x}}^*(\mathbf{x})$ (memoization) by:

$$\begin{aligned}&G(\mathbf{x} \in \mathcal{K}^N) = 0 \\ &G(\mathbf{x} \in \mathcal{K}^i) = \min_{\bar{\mathbf{x}} \in \mathcal{K}^{i+1}} \left\{ l(\bar{\mathbf{x}} - \mathbf{x}) + G(\bar{\mathbf{x}}) \mid \bar{\mathbf{x}} - \mathbf{x} \in \mathcal{U}_a \right\} \\ &\bar{\mathbf{x}}^*(\mathbf{x} \in \mathcal{K}^i) = \arg\min_{\bar{\mathbf{x}} \in \mathcal{K}^{i+1}} \left\{ l(\bar{\mathbf{x}} - \mathbf{x}) + G(\bar{\mathbf{x}}) \mid \bar{\mathbf{x}} - \mathbf{x} \in \mathcal{U}_a \right\}\end{aligned} \tag{21}$$

The minimum cost $J_{min} = \min_{\mathbf{x}}\{G(\mathbf{x} \in \mathcal{K}^0)\}$ can be found as soon as the graph is fully connected, and the memoization can be used to recover the optimal trajectory. The whole procedure is summarized in Algorithm 1 (MoboConTP).

For fast computation, we use coarse step sizes for Algorithm 1 then apply linear interpolation to match with controller's rate, so the trajectory is piecewise C^1-continuous.

D. Completeness and Optimality

Theorem 1 (Completeness): Algorithm 1 only reports `Infeasible` when there is indeed no feasible trajectory.

Proof: Since Algorithm 1 only reports `Infeasible` when it runs into $\mathcal{K}^i = \emptyset$, we can prove by contradiction that: if there exists a feasible trajectory $\{\mathbf{x}^0, ..., \mathbf{x}^N\}$, then $\mathcal{K}^i$ contains $\mathbf{x}^i$ for all $i \in [0, N]$. Using backward induction:

- Initialization: $\mathbf{x}^N \in \mathcal{K}^N$ by construction.
- Induction: Assume that $\mathbf{x}^{i+1} \in \mathcal{K}^{i+1}$; Since $\mathbf{x}^i$ and $\mathbf{x}^{i+1}$ are in a feasible trajectory, they satisfy $\mathbf{x}^{i+1} - \mathbf{x}^i \in \mathcal{U}_a$, which means $\mathbf{x}^i \in \mathcal{Q}(\mathcal{K}^{i+1}) = \mathcal{K}^i$ according to (18), (20).

By induction, we get $\mathbf{x}^i \in \mathcal{K}^i \; \forall i \in [0, N]$, which means that Algorithm 1 will not report `Infeasible` if a feasible trajectory exists.

Theorem 2 (Optimality): If Algorithm 1 returns an output trajectory, then it is indeed the optimal trajectory.

Proof: Firstly, if Algorithm 1 returns a sequence $\{\mathbf{x}^0, ..., \mathbf{x}^N\}$, one can show by forward induction that this sequence is a feasible trajectory. Secondly, Algorithm 1 tracks the minimum cost-to-go from each node to the goal stage in the same way as dynamic programming, so by the principle of optimality [11], the returned sequence has the minimum cost-to-go from stage 0 to N, which means it is indeed the optimal trajectory.

Algorithm 1: MoboConTP - optimal base trajectory planning for end-effector trajectory continuity

Input: Discretized admissible base configuration spacetime $\mathcal{X}_a = \{\mathcal{X}_a^0, ..., \mathcal{X}_a^N\}$, set of admissible controls $\mathcal{U}_a$
Output: Optimal trajectory $\{\mathbf{x}^0, ..., \mathbf{x}^N\}$

Initialization: $\mathcal{K}^N \leftarrow \mathcal{X}_a^N$ and $G(\mathbf{x}) \leftarrow 0 \; \forall \mathbf{x} \in \mathcal{K}^N$
`/* Backward iterations */`
for $i \in [N-1, ..., 0]$ **do**
  $\mathcal{K}^i \leftarrow \emptyset$
  for $\mathbf{x} \in \mathcal{X}_a^i$ **do**
    $ValidNode \leftarrow False$; $G(\mathbf{x}) \leftarrow \infty$
    for $\bar{\mathbf{x}} \in \mathcal{K}^{i+1}$ *such that* $\bar{\mathbf{x}} - \mathbf{x} \in \mathcal{U}_a$ **do**
      $ValidNode \leftarrow True$
      if $l(\bar{\mathbf{x}} - \mathbf{x}) + G(\bar{\mathbf{x}}) < G(\mathbf{x})$ **then**
        $G(\mathbf{x}) \leftarrow l(\bar{\mathbf{x}} - \mathbf{x}) + G(\bar{\mathbf{x}})$
        $\bar{\mathbf{x}}^*(\mathbf{x}) \leftarrow \bar{\mathbf{x}}$
    if *ValidNode* **then**
      $\mathcal{K}^i.Insert(\mathbf{x})$
  if $\mathcal{K}^i = \emptyset$ **then**
    return `Infeasible`
`/* Recover the optimal trajectory */`
$\mathbf{x}^0 \leftarrow \arg\min_{\mathbf{x}} G(\mathbf{x} \in \mathcal{K}^0)$
for $i \in [0, ..., N-1]$ **do**
  $\mathbf{x}^{i+1} \leftarrow \bar{\mathbf{x}}^*(\mathbf{x}^i)$

V. Evaluations

Our mobile manipulator consists of a DENSO VS-087 6-DOF arm mounted on a Clearpath Ridgeback 3-DOF omnidirectional mobile base. The following experiment and simulations illustrate our method in mobile printing at different difficulty levels, all were performed using CPU AMD 5900HX with 16GB RAM, running ROS Melodic in Ubuntu 18.04. Simulations were performed in OpenRAVE environment [28]. A hardware demo and a simulation example are shown in the accompanying video (https://youtu.be/yyBv3xGClnk).

A. Method visualization: Mobile 1D printing

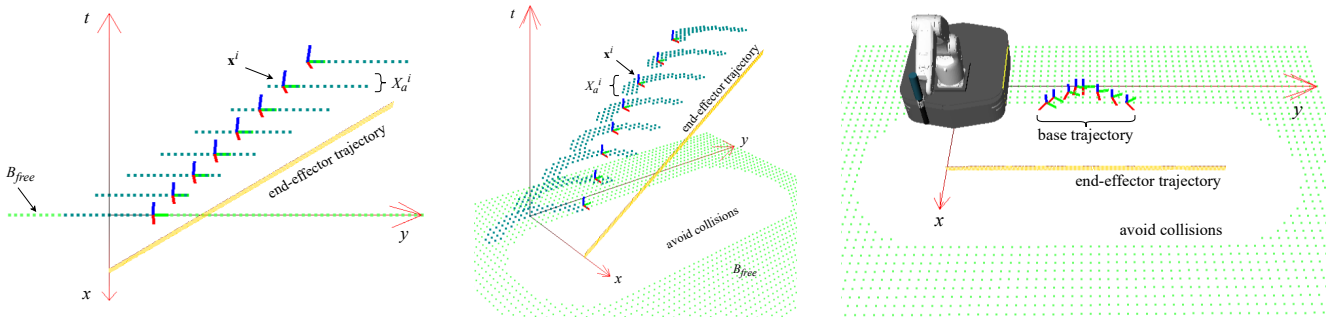
Fig. 5 shows the optimal base trajectory solutions for 3 cases of mobile base in 1D printing a horizontal line. Since the B-spacetime for 3D base is 4-dimensional (t, x, y, φ), Fig. 5c can only show the solution trajectory without time axis.

B. Algorithm benchmarking: Mobile 3D printing

Next, we benchmark MoboConTP algorithm and compare it with a baseline (Dijkstra's algorithm) in a 3D printing task as shown in Fig. 1: an NTU shape with the total printing path length of $d = 112.9m$ (10 layers). Table I compares the computation time of two methods (both using $\Delta t = 2.5s$, $\Delta v_x = \Delta v_y = 5cm/s$, $\Delta\omega = \pi/30\,rad/s$). MoboConTP is faster by running optimization and graph construction concurrently.

TABLE I
Planning time comparison (in NTU test, Fig. 1)

Planning time for:	8 layers	10 layers	12 layers	14 layers
MoboConTP	89.9s	108.4s	129.7s	155.1s
Baseline (Dijkstra)	120.3s	155.2s	184.4s	515.0s



(a) 1D mobile base ($x = \varphi = 0$) (b) 2D mobile base ($\varphi = 0$) (c) Omnidirectional 3D planar mobile base

Fig. 5. Mobile 1D printing a $2.1m$-long line with different base's DOFs. (a) and (b) are seen in B-spacetime, (c) is seen in task space.

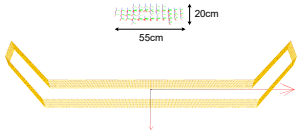


Fig. 6. Optimal base trajectory for mobile 3D printing task in [3]

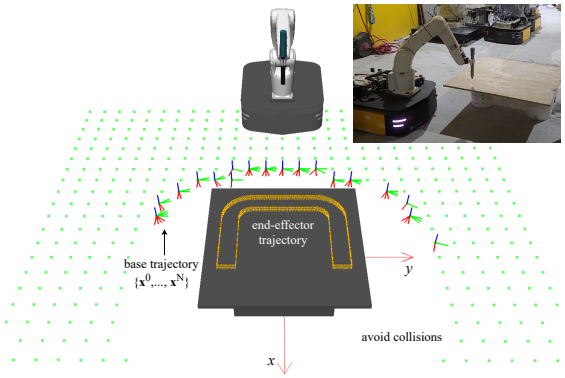
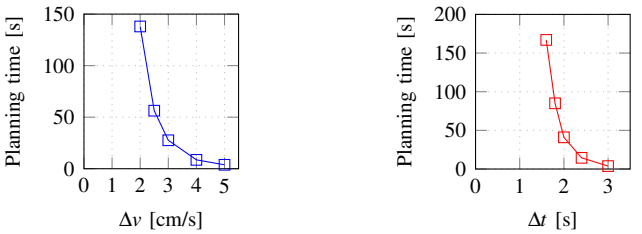


Fig. 7. Hardware mobile 3D printing demo: U shape ($0.9 \times 0.675 \times 0.05m$, 5 layers, total printing path: $19.85m$) with constant nozzle speed of $10cm/s$.



(a) Planning time for U shape (5 layers), using $\Delta t = 3s$

(b) Planning time for U shape (5 layers), using $\Delta v = 5cm/s$

Fig. 8. Complexity of Algorithm 1 with respect to discretization step sizes

C. Comparison: Mobile 3D printing

We would like to compare our method with manual planning in [3]. Fig. 6 shows our solution where the mobile base moves in a region $55 \times 20cm$ instead of $80 \times 10cm$, with an average speed $2cm/s$ which is lower than $3.5cm/s$ in [3].

D. Hardware demo: Mobile 3D printing

Fig. 7 shows a hardware demo of a 3D printing task (5 layers of U shape) with total printing path length $d = 19.85m$. Our algorithm planned an optimal mobile base trajectory in $3.8s$ (using $\Delta t = 3s$, $\Delta v_x = \Delta v_y = 5cm/s$, $\Delta\omega = \pi/30\, rad/s$).

E. Complexity analysis

1) Complexity with respect to discretization step sizes: Number of stages is $N \propto 1/\Delta t$; number of nodes per stage is $|\mathcal{X}_a^i| \propto (1/\Delta x \Delta y \Delta\varphi) \propto (1/\Delta t^3 \Delta v_x \Delta v_y \Delta\omega)$. Based on Algorithm 1, we expect its complexity to be no higher than $O\left((1/\Delta t)^7 (1/\Delta v_x \Delta v_y \Delta\omega)^2\right)$. Fig. 8 shows in our test case that, as $\Delta v := \Delta v_x = \Delta v_y$ and Δt decrease, the computation time increases polynomially with order $O((1/\Delta v)^{3.9})$ and $O((1/\Delta t)^{6.0})$ respectively (based on log-log analysis).

2) Complexity with respect to task length: Table I shows that the computation time of Algorithm 1 increases linearly, since $N \propto d$. This is crucial for large-scale applications.

VI. Conclusion

In this paper, we have proposed a method for solving the mobile manipulator motion planning problem under end-effector trajectory continuity constraint, subject to other constraints and base trajectory optimization. We have also presented a complete and optimal algorithm to plan optimal base trajectories, which was evaluated in several mobile printing tasks. Our future directions include: integrating planning with control for actual mobile concrete printing and accuracy evaluations, and addressing the following limitations:

- The joint kinematic constraints were not considered.
- The base acceleration constraints were not considered.
- During manipulator trajectory planning, although continuous joint trajectories were obtained in our tests, this may not be guaranteed for other tasks or robots.
- The base trajectory planning algorithm can be improved using other trajectory optimization methods.
- Infeasible tasks can be tackled by a multi-robot team.

Acknowledgment

This research was supported by the National Research Foundation, Prime Minister's Office, Singapore under its Medium-Sized Centre funding scheme, CES_SDC Pte Ltd, and Sembcorp Architects & Engineers Pte Ltd.

REFERENCES

[1] J. Sustarevas, D. Kanoulas, and S. Julier, "Autonomous mobile 3d printing of large-scale trajectories," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 6561–6568.

[2] J. Sustarevas, D. Kanoulas, and S. Julier, "Task-consistent path planning for mobile 3d printing," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 2143–2150.

[3] M. E. Tiryaki, X. Zhang, and Q.-C. Pham, "Printing-while-moving: a new paradigm for large-scale robotic 3d printing," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 2286–2291.

[4] X. Zhang, M. Li, J. H. Lim, Y. Weng, Y. W. D. Tay, H. Pham, and Q.-C. Pham, "Large-scale 3d printing by a team of mobile robots," *Automation in Construction*, vol. 95, pp. 98–106, 2018.

[5] K. Dörfler, G. Dielemans, L. Lachmayer, T. Recker, A. Raatz, D. Lowke, and M. Gerke, "Additive manufacturing using mobile robots: Opportunities and challenges for building construction," *Cement and Concrete Research*, vol. 158, p. 106772, 2022.

[6] R. Malhan and S. K. Gupta, "Finding optimal sequence of mobile manipulator placements for automated coverage planning of large complex parts," in *Proceedings of the ASME 2021 International Design Engineering Technical Conferences (IDETC)*, 2022.

[7] J. Xu, Y. Domae, T. Ueshiba, W. Wan, and K. Harada, "Planning a minimum sequence of positions for picking parts from multiple trays using a mobile manipulator," *IEEE Access*, vol. 9, pp. 165 526–165 541, 2021.

[8] Q.-N. Nguyen, N. Adrian, and Q.-C. Pham, "Task-space clustering for mobile manipulator task sequencing," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3693–3699.

[9] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.

[10] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 639–646.

[11] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[12] S. E. Jenny, L. L. Pietrasik, E. Sounigo, P.-H. Tsai, F. Gramazio, M. Kohler, E. Lloret-Fritschi, and M. Hutter, "Continuous mobile thin-layer on-site printing," *Automation in Construction*, vol. 146, p. 104634, 2023.

[13] K. Nagatani, T. Hirayama, A. Gofuku, and Y. Tanaka, "Motion planning for mobile manipulator with keeping manipulability," in *IEEE/RSJ international conference on intelligent robots and systems*, vol. 2. IEEE, 2002, pp. 1663–1668.

[14] G. Oriolo, M. Ottavi, and M. Vendittelli, "Probabilistic motion planning for redundant robots along given end-effector paths," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2. IEEE, 2002, pp. 1657–1662.

[15] G. Oriolo and C. Mongillo, "Motion planning for mobile manipulators along given end-effector paths," in *Proceedings of the 2005 IEEE international conference on robotics and automation*. IEEE, 2005, pp. 2154–2160.

[16] T. Welschehold, C. Dornhege, F. Paus, T. Asfour, and W. Burgard, "Coupling mobile base and end-effector motion in task space," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.

[17] J. Haviland, N. Sünderhauf, and P. Corke, "A holistic approach to reactive mobile manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3122–3129, 2022.

[18] T. Sandakalum and M. H. Ang Jr, "Motion planning for mobile manipulators—a systematic review," *Machines*, vol. 10, no. 2, p. 97, 2022.

[19] G. B. Avanzini, A. M. Zanchettin, and P. Rocco, "Constraint-based model predictive control for holonomic mobile manipulators," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1473–1479.

[20] J. Pankert and M. Hutter, "Perceptive model predictive control for continuous mobile manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6177–6184, 2020.

[21] M. Giftthaler, F. Farshidian, T. Sandy, L. Stadelmann, and J. Buchli, "Efficient kinematic planning for mobile manipulators with non-holonomic constraints using optimal control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3411–3417.

[22] F. Zacharias, C. Borst, M. Beetz, and G. Hirzinger, "Positioning mobile manipulators to perform constrained linear trajectories," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 2578–2584.

[23] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1970–1975.

[24] Z. Xian, P. Lertkultanon, and Q.-C. Pham, "Closed-chain manipulation of large objects by multi-arm robotic systems," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1832–1839, 2017.

[25] D. Rakita, B. Mutlu, and M. Gleicher, "Stampede: A discrete-optimization method for solving pathwise-inverse kinematics," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3507–3513.

[26] D. Rakita, B. Mutlu, and M. Gleicher, "Relaxedik: Real-time synthesis of accurate and feasible robot arm motion." in *Robotics: Science and Systems*, vol. 14. Pittsburgh, PA, 2018, pp. 26–30.

[27] M. Trümper, "Lagrangian mechanics and the geometry of configuration spacetime," *Annals of Physics*, vol. 149, no. 1, pp. 203–233, 1983.

[28] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, The Robotics Institute Pittsburgh, 2010.

[29] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, 2018.