TRACE:
Interactive Bi-Directional Cable Tracing Amid Clutter

Anonymous Authors

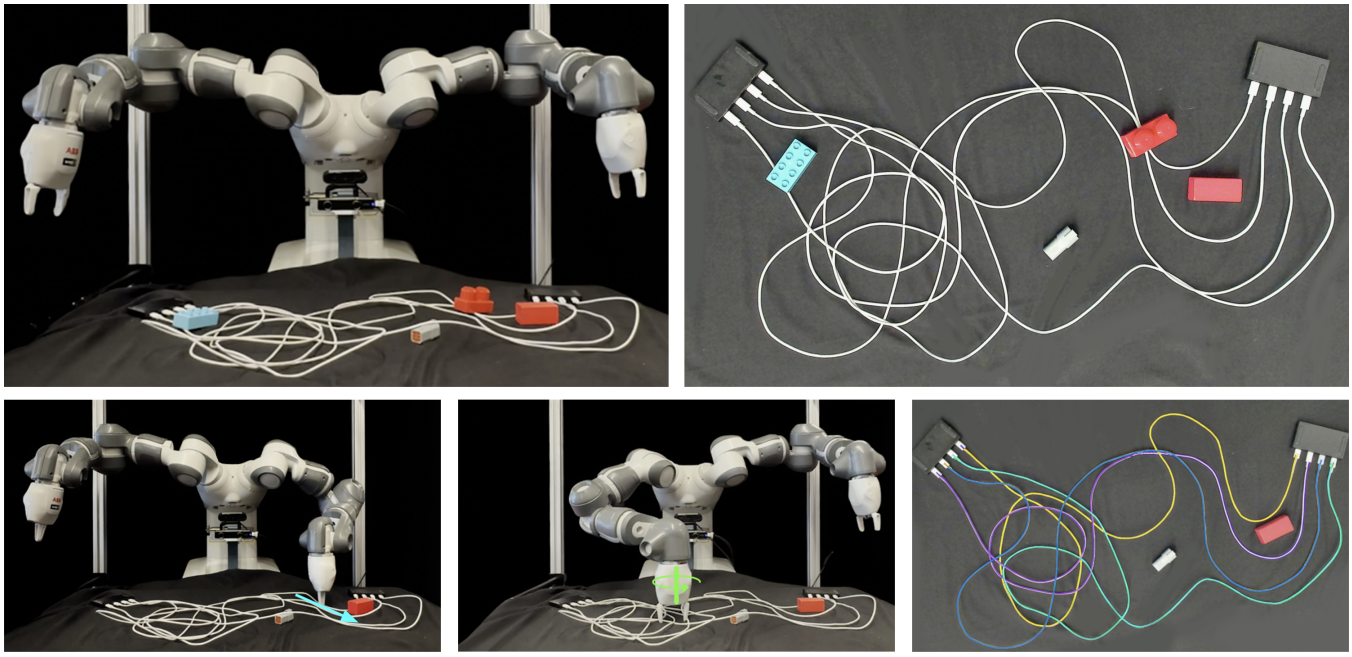

Fig. 1: Overview of the TRACE framework. (Top left): Initial scene with 4 cables and 4 foreground clutter objects. (Top right): Overhead image of the initial scene as input to TRACE. The novel interactive perception primitives used to resolve visual ambiguities are (bottom left): Divergence Push and (bottom center): Cluster Dilation. Finally, (bottom right) shows a trace of 100% of cables in the scene.

***Abstract*— Accurate state estimation (tracing) of deformable cables is a critical challenge across manufacturing, construction, homes, and surgery, where precise cable management directly impacts operational safety and efficiency. However, resolving the state of multiple cluttered and tangled cables poses challenges due to occlusions, overlap, and ambiguous crossings. We introduce Two-way Routing And Cable Estimation (TRACE), which combines bi-directional cable tracing with novel interactive perception primitives—Divergence Push and Cluster Dilation—to actively resolve ambiguities. Evaluations on 110 physical experiments suggest that TRACE increases the percentage of cable length correctly traced from ∼60% to ∼90% in complex scenarios (with up to 4 cables and 40 overlapping crossings) over previous tracing methods. Project website is linked at: `https://trace-icra.github.io/`.**

I. Introduction

The manipulation and perception of Deformable Linear Objects (DLOs), such as cables, hoses, and threads, pose significant challenges in applications such as electrical wiring, maintenance, product assembly, surgery, and construction. In many modern industrial facilities, cluttered and tangled cables can reduce reliability, compromise safety, and complicate troubleshooting efforts. Unlike rigid objects, cables exhibit complex deformations, self-occlusions, and entanglements, which make them difficult to trace.

Single-cable tracing methods have shown promising results in controlled environments, using both analytical models based on geometric structures (V. Viswanath *et al.* [2]) and data-driven techniques that trace cables from visual input (K. Shivakumar *et al.* [3]). Among these, Heterogeneous Autoregressive Learned Deformable Linear Object Observation and Manipulation (HANDLOOM 1.0) [4] is an RGB vision-based, autoregressive framework that sequentially traces cables by predicting the most likely continuation path using learned visual features and contextual cues. While HANDLOOM 1.0 successfully traces individual cables in most structured environments, its extension to multi-cable scenarios faces significant challenges.

Interactive Perception allows robots to actively manipulate the environment to minimize state uncertainty and enhance observability (J. Bohg *et al.* [5]). By physically interacting with object clutter and cables, robots can reveal occluded segments and resolve ambiguities at cluttered crossings. The Modular Architecture for Integrating Interactive Perception (MANIP) framework (J. Yu *et al.* [1]) demonstrated that modular system architectures incorporating learned and model-based manipulation primitives can significantly improve single-cable estimation in the presence of multiple cables. Using the MANIP framework, HANDLOOM 2.0 applied interactive perception to improve tracing success rates by up to 88%, but only traced 60% to 68% of cable length correctly in configurations with 3 or 4 cables.

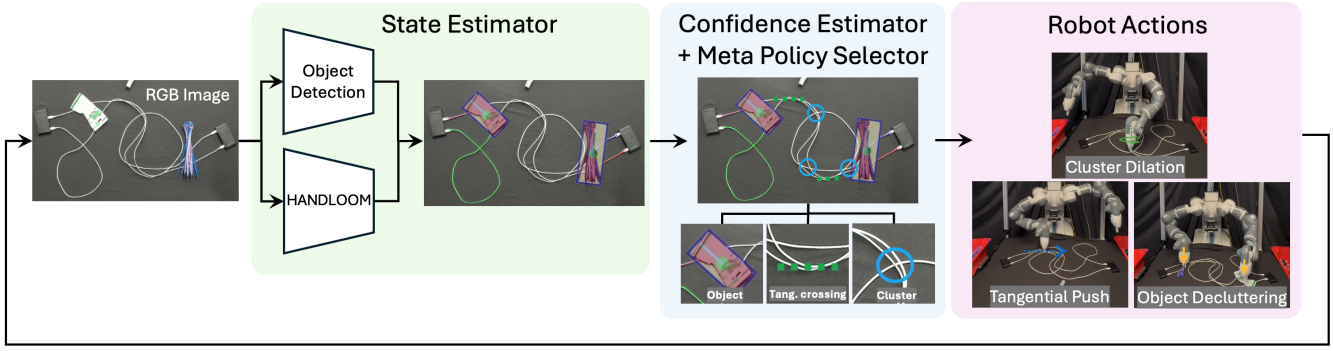

Fig. 2: Overview of the TRACE architecture using the MANIP [1] framework. The system first processes the overhead RGB image to detect occlusions by comparing the TRACE-predicted cable state with object masks. If an occlusion is detected, an object decluttering primitive is executed. Then, divergence points are identified via bi-directional tracing and classified as tangential crossings or cable clusters. This prompts the Meta Policy Selector to activate the corresponding interactive perception primitive.

In this work, we introduce TRACE, a framework that extends previous multi-cable tracing approaches by introducing bi-directional tracing and new interactive perception primitives to systematically resolve ambiguities caused by occlusions, crossings, and high-density cable arrangements. Unlike HANDLOOM 2.0, which relies on predefined heuristics (entropy, local cable density, trace length, and cable connector locations) to trigger intervention actions, TRACE globally analyzes cable connectivity to identify points of uncertainty and enforce consistency across multiple traces. TRACE's cable disambiguation capabilities are shown in Figure 1, and its architecture is further visualized in Figure 2. This paper makes the following contributions:

1) The Two-way Routing And Cable Estimation (TRACE) framework for robotic multi-cable tracing in cluttered environments, with a novel monocular bi-directional cable tracing algorithm.
2) Two novel task-oriented robot interactive perception primitives (Divergence Push and Cluster Dilation) combining geometric information and Hessian-based disambiguation along with bimanual foreground decluttering.
3) Physical evaluations on 110 unique real-world cluttered cable scenarios that suggest significant performance improvements over HANDLOOM 2.0 and RT-DLO [6].

II. Related Work

Early vision-only single cable tracing methods enforced spline smoothness [7–10] and registered visual features across frames (W. H. Lui *et al.* [11], T. Tang *et al.* [12]). More recent methods expand to multiple cables and self-intersecting cables (J. Xiang *et al.* [13, 14]). While effective for isolated, well-lit cables, these analytical pipelines degrade when multiple cables overlap. More recently, learning-based approaches have gained traction for predicting cable properties (A. Caporali *et al.* [15]) by using instance segmentation models (S. Zhaole *et al.* [16]). For example, RT-DLO [6] extracts cable structures using skeletonization from semantic masks with graph-based topology, while M. Yan *et al.* [17] uses a coarse-to-fine iterative refinement strategy by progressively expanding the region of interest. Concurrently, K. Lv *et al.* [18] learns a two-branch PointNet++ network to estimate the 3-D state of a single segmented deformable linear object behind foreground occlusions. HANDLOOM 1.0 [4] uses a probabilistic learning-based approach that autoregressively predicts trace points using a UNet convolutional network applied to locally cropped, reoriented overhead images. Despite its effectiveness in structured environments, HANDLOOM 1.0 struggles in cluttered multi-cable scenes where occluding objects and high cable density lead to failures in complex industrial settings. Unlike these prior works, TRACE scales to higher complexity, handling longer cables with more diverse configurations, including those with up to 40 overlapping crossings. In addition, TRACE operates on monocular RGB images alone; it does not rely on depth data to disambiguate occlusions.

To improve tracing performance in ambiguous scenarios, interactive perception has been explored to resolve occlusions and crossings [5]. For instance, SGTM 2.0 [3] combined an analytical cable tracing method with interactive manipulation primitives, demonstrating that targeted interactions significantly improve untangling. Building on these insights, HANDLOOM 2.0 [1] extends interactive perception to the cable state estimation task by integrating HANDLOOM 1.0 with manipulation policies for intervention. However, determining where and when to intervene via interactive perception in cable disambiguation scenes remains an open challenge, particularly as the complexity of multi-cable configurations increases. TRACE handles complex cable configurations with up to 5 tangential crossings and 40 total crossings per scene, high-density clusters that cover up to 90% of a 1 in^2 area with cables, and up to 4 foreground objects cluttering a scene. TRACE refines intervention strategies based on local and global cable geometry, and demonstrates transferable improvement in trace performance from a baseline evaluation against HANDLOOM 2.0 and RT-DLO.

III. Problem Formulation

TRACE makes the following assumptions:

1) The tracing algorithm is only provided RGB images: no depth maps, stereo baselines, or calibrated range data.
2) Cables are visually distinguishable from a black background.

3) Foreground clutter, or objects that lie on top of cables (versus background clutter, or patterns that lie underneath cables), have a minor principal axis length smaller than the width of the robot gripper.
4) All cable endpoints (i.e. connectors) are inserted into USB hubs.

The workspace is defined as a planar region $\mathcal{W} \subset \mathbb{R}^2$, containing $n \leq 4$ white cables. Each cable i is represented in the xy-plane by a continuous centerline function $\theta_{i,t}(s)$, which gives the true (x, y) position of cable i at timestep $t \in \{0, 1, 2, ..., T_{max}\}$ at normalized arc-length parameter $s \in [0, 1]$ from its first connector e. The complete estimated cable state for all cables at timestep t is given by $\hat{\Theta}_t = \bigcup_{i=1}^{n} \hat{\theta}_{i,t}(s)$, where $\hat{\theta}_{i,t}(s)$ is an estimate of $\theta_{i,t}(s)$. At each timestep t, the system receives a monocular RGB image observation $\mathbf{I_t}$ from a single overhead camera (no depth channel). The objective is to accurately reconstruct the complete ground truth cable state Θ_t from visual input $\mathbf{I_t}$ and identify the corresponding connectors of each cable i. This requires generating cable state representations that avoid topological inconsistencies such as mismatched connectors or erroneous trace paths due to crossings between distinct cables.

In cases where visual ambiguities, occlusions, or dense cable configurations make passive observation insufficient, TRACE uses robotic interactive perception primitives to actively manipulate the workspace. At each timestep t, the robot applies an interactive perception primitive $\mathbf{a_t} \in \mathcal{A}$, where $\mathcal{A}$ is the set of implemented primitives, producing an updated true cable state Θ_{t+1} and updated estimate $\hat{\Theta}_{t+1}$.

IV. Method

A. Connector Detection

The system first identifies (but does not attempt to match) all cable connectors to initialize the tracing algorithm. We collected a dataset of labeled images containing random connector configurations to supervise a Faster R-CNN model [19] to detect the boundary boxes of potential connectors, following similar approaches from previous work [2, 20]. Given an image observation $\mathbf{I_t}$ at timestep t, the model outputs a set of detected connectors $\mathcal{E}(\mathbf{I_t}) = \{e_1, e_2, \ldots, e_{2n}\}$, where each $e_i \in \mathbb{R}^2$ is the estimated pixel location of a connector.

B. Bimanual Decluttering

Occlusions from external objects in the workspace can introduce ambiguity and lead to errors in cable tracing. To address this, we introduce an additional manipulation primitive, *Object Decluttering*, which removes external objects that obstruct the tracing process. Given an image observation $\mathbf{I_t}$, an object segmentation model identifies non-cable objects in the scene, then the HANDLOOM 1.0 [4] tracer reconstructs cable paths. When an object is detected along a cable path, a pick-and-place action is executed to remove the obstruction. If two objects are detected on opposite sides of the scene, the system takes advantage of the bimanual capabilities of the robot to simultaneously grasp and clear away both objects.

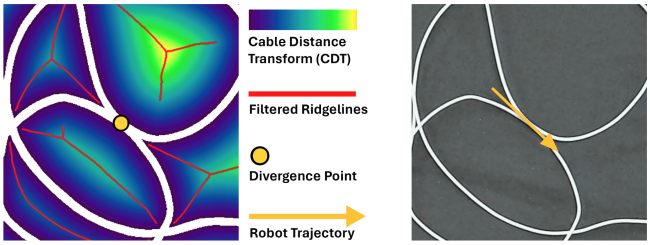

Fig. 3: (Left): the **Cable Distance Transform (CDT)** maps each pixel value to the Euclidean distance between the pixel and the nearest cable (white). Ridgelines (red) are from the CDT using a Hessian-based filtering algorithm and highlight potential paths for robot trajectory planning while minimizing unintended cable interaction. Divergence points (circled) indicate ambiguous intersections, resolved through targeted push primitives. (Right): a **Divergence Push** primitive starts along a ridgeline at one end and moves through the divergence point to attempt separation at tangential crossings.

C. Bi-Directional Tracing

TRACE uses the HANDLOOM 1.0 tracer [4] as the single-cable tracing algorithm with an additional termination condition in the case of foreground clutter, where the system detects clutter objects that obstruct the cable trace which necessitate removal.

To systematically identify ambiguous regions, TRACE performs bi-directional tracing by initiating a cable trajectory from each connector e_i, so there will be twice as many traces as cables. In a correctly traced scenario, each pair of traces will align and overlap. This bi-directional process produces two independent estimates for each cable, starting from opposing connectors. By aggregating these traces into a unified state estimate, the system can identify discrepancies where the two traces fail to connect, exhibit inconsistencies, or fail to overlap to form a single continuous trajectory. When cables overlap in a tangential crossing, such as in Figure 3, or are closely packed as in Figure 5, visual ambiguities arise that make it difficult to distinguish individual paths.

Divergence Points occur when contradictory or incomplete connector assignments are present (many-to-one or one-to-none mappings), indicating that a cable trace $\hat{\theta}_i$ intersects another trace $\hat{\theta}_j$. To identify and disambiguate divergence points, at each timestep t, TRACE executes in two phases:

Phase 1: Reciprocal Consistency Verification: TRACE first identifies regions of ambiguity in the scene by assessing reciprocal consistency—i.e., whether the forward and backward traces from each cable's connectors overlap. Traces that lack a valid reciprocal match (e.g., traces for one cable that diverge into two separate paths, or traces that terminate prematurely) create a point of uncertainty along their paths. These points are labeled as potential divergence points and passed to the next analysis stage for further classification and refinement.

Phase 2: Divergence Point Analysis: TRACE then searches for candidate divergent trace pairs using a hierarchical approach. For incompletely traced cables, it first attempts direct matching by searching for another trace whose connector corresponds to the starting connector of the incomplete trace. If this method fails, TRACE computes the overlap between the incomplete trace and all other traces, selects the pair with the highest overlap, and determines the divergence

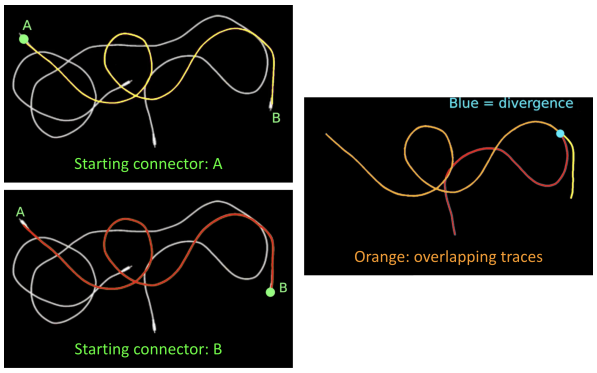

Fig. 4: TRACE's bi-directional tracing that identifies divergence points on 2 cables. (Left): cable traces initialized starting from 2 connectors of the same cable. (Right): overlapping cable traces are shown in orange; a divergence point in blue arises when the traces diverge into two non-overlapping paths.

point d_i as the location where the two traces transition from overlapping to non-overlapping, as shown in Figure 4. Divergence points are classified based on local cable density $\rho(d_i)$, where $\rho(d_i)$ counts the number of distinct cables passing within a small radius r around d_i. A *cable cluster*, shown in Figure 5, occurs when $\rho(d_i)$ exceeds a threshold τ_c, indicating high-density regions where cables visually merge, increasing the likelihood of erroneous connector assignments. In contrast, a *tangential crossing* arises when $\rho(d_i) < \tau_c$, representing cases where cables intersect at shallow angles, creating momentary overlaps without persistent occlusions or entanglements. The threshold τ_c is manually defined based on empirical observations of visually ambiguous scenarios where distinguishing between overlapping and non-overlapping traces is challenging. Formally, we define the following:

$$\mathcal{K} = \{d_i \mid \rho(d_i) \geq \tau_c\}, \quad \mathcal{T} = \{d_i \mid \rho(d_i) < \tau_c\} \tag{1}$$

where $\mathcal{K}$ represents the set of divergence points d_i where cable density is above the threshold τ_c. and $\mathcal{T}$ represents the set of tangential crossings.

D. Two Novel Interactive Perception Primitives

We introduce two new interactive perception primitives: *Cluster Dilation*, which physically separates densely packed cables located near trace points in set $\mathcal{K}$, and *Divergence Push*, which attempts to disambiguate tangential crossing points in set $\mathcal{T}$. An interactive perception primitive similar to the Divergence Push was first introduced in HANDLOOM 2.0. In TRACE, we enhance and generalize this method by incorporating bi-directional traces.

For robot trajectory planning in a multi-cable environment, we introduce the Cable Distance Transform (CDT), illustrated by Figure 3. The CDT, related to the Voronoi diagram [21], is an adaptation of the Euclidean Distance Transform [22]. Given a binary mask generated from cable traces, CDT is defined as $D_c(p) = \min_{q \in \mathcal{C}} \|p - q\|$, where $D_c(p)$ computes the shortest Euclidean distance from pixel p to the nearest cable pixel $q \in \hat{\Theta}_t$. The CDT maps open areas within densely packed cable environments, guiding the robot to initiate push interactions that reduce unintended cable disturbance.

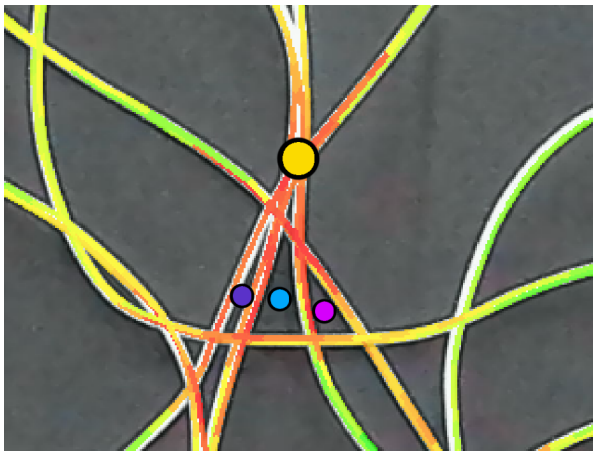

Fig. 5: **Cable Density and Cluster Dilation.** The cables are overlaid with a density heatmap, where red indicates high local cable density. The yellow dot marks a divergence point classified as a cluster due to the high-density region. The three dots below represent the centroids of candidate open regions; the cyan (center) dot corresponds to the open region with the largest pixel area selected for a Cluster Dilation.

Cluster Dilation: To separate clusters illustrated by Figure 5, TRACE finds open regions near each $d_i \in \mathcal{K}$. Open regions are defined as non-cable pixels that are sufficiently distant from any cable, denoted $\mathcal{O} = \{p \in \mathbf{I_t} \mid D_c(p) > \tau_o\}$, where τ_o is a separation threshold. To improve robustness, TRACE uses area constraints to filter out open regions that are either too small or large for a gripper insertion to be effective. Let $A(\mathcal{O})$ denote the pixel area of an open region R . We define the filtered set of open regions as: $\mathcal{O}' = \{\mathcal{O} \mid \alpha_{min} \leq A(\mathcal{O}) \leq \alpha_{max}\}$, where α_{min} and α_{max} are the minimum and maximum area thresholds respectively. The system then finds the open region $R \in \mathcal{O}'$ with the largest pixel area, moves a closed gripper to $p^* = \text{argmax}_{p \in R} D_c(p)$, fully opens the gripper, and applies two 180° rotations in opposite directions in an attempt to separate the cables.

Divergence Push: In the CDT, pronounced local regions of high linearity appear within the distance transform which correspond to regions of maximum local distance change. These occur where two or more cable pixels are equidistant from the queried pixel, forming natural boundaries that separate the influence zones of each cable. Consequently, these ridges map the medial axis between neighboring cables, delineating the safest paths for robot trajectory planning. TRACE performs ridge detection on the CDT using the Frangi vesselness filter [23] shown in Figure 3 as red lines. This filter analyzes the second-order local structure of the CDT by computing the eigenvalues of the Hessian matrix, which highlights the structures corresponding to ridges. The resulting ridge set $\mathcal{R}$ is defined as: $\mathcal{R} = \{p \mid V(p) > \tau_r\}$, where τ_r is a threshold to distinguish significant ridges from the background. The vesselness measure $V(p)$ quantifies the likelihood that a pixel p belongs to a ridge-like structure by evaluating the ratio and magnitude of the eigenvalues. High vesselness values correspond to regions where the Hessian eigenvalues indicate elongated linear features, which naturally align with the medial axes between cables.

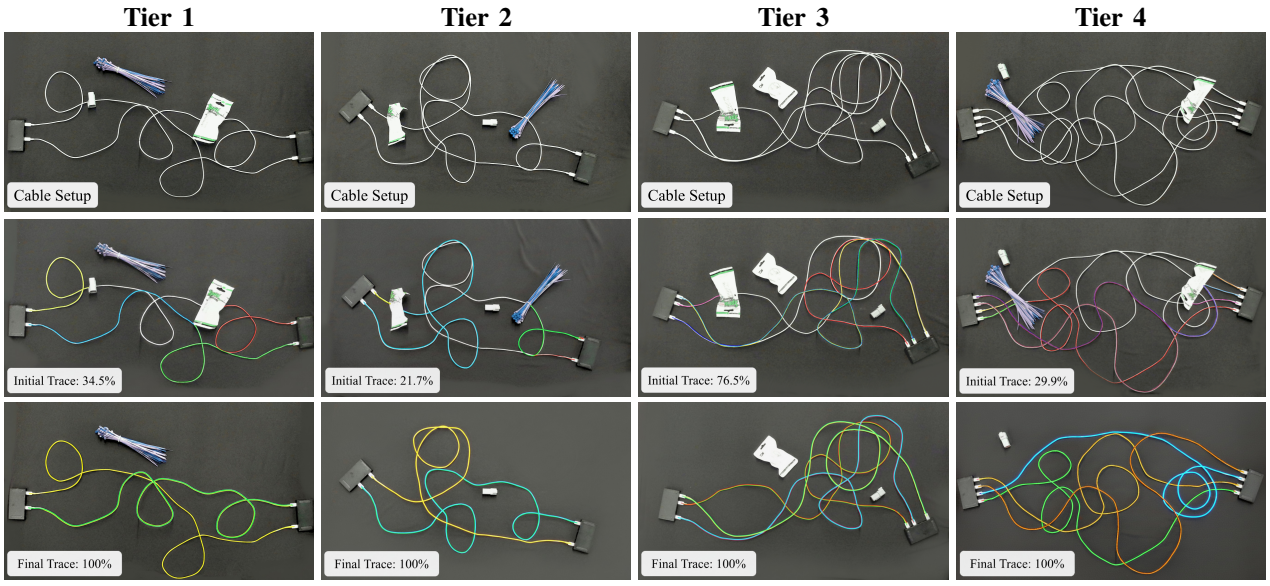

Fig. 6: Four examples; each column represents a different tier. The first row shows the initial scene with cluttered cables and external objects, the second row displays the initial predicted traces with occlusions, the third row shows the completed traces after object decluttering and interactive perception primitives.

In practice, following these ridges often reduces unintended cable interaction. To determine the push direction, TRACE computes the tangential lines of the crossing cables. The push interaction starting point is then initialized along the ridge set $\mathcal{R}$, constrained by the two tangential lines and a predefined keep-out distance of 12 pixels from nearby cables. The push action is executed along this ridge: beginning at the push interaction starting point, sliding the end-effector through the divergence point, and continuing for an equal distance beyond the divergence point.

E. Pipeline and Termination

We formalize the complete pipeline of TRACE. At each timestep t, the system receives an image observation $\mathbf{I_t}$ and detects external objects and cable connectors $\mathcal{E}(\mathbf{I_t}) = \{e_1, e_2, \ldots, e_n\}$ using a trained connector detection model. Each detected connector e_i initializes the HANDLOOM tracer, producing an estimated trajectory $\hat{\theta}_i$. After tracing, the system identifies errors that may affect trajectory reconstruction.

If any external objects actively occlude the cable trace, the system applies an object removal action. It then identifies divergence points ($\mathcal{D}_t$), classifying them as either tangential crossings ($\mathcal{T}_t$) or high-density cable clusters ($\mathcal{K}_t$). Then, the system applies the associated interactive perception action $\mathbf{a_t}$ to update the estimated complete cable state $\hat{\Theta}_{t+1}$, improving visibility and refining trajectory estimates $\hat{\theta}_{i,t+1}$. This process iterates until the termination condition $T_{max} = 10$ timesteps is reached or until all ambiguities are resolved such that the reconstructed cable set $\hat{\Theta} = \bigcup_{i=1}^{n} \hat{\theta}_i$ is topologically consistent.

V. Experiments

Hardware: The experimental setup consists of a bimanual ABB YuMi robot with motion planning trajectories solved by Jacobi Motion [24], and an overhead Logitech BRIO RGB camera positioned 1 meter above the workspace. The robot operates over a planar workspace, where 2-4 standard white 6-foot USB-C to USB-C cables are randomly arranged. The connectors of each cable are plugged into two 8x5 cm black USB hubs on opposite sides of the workspace. The overhead camera captures 4K resolution images of the cable environment.

TABLE I: Tiers of complexity for evaluation

	Tier 1	Tier 2	Tier 3	Tier 4
# Cables	2	2	3	4
# Tangential Crossings	2	3	3-4	4-5
# Foreground Objects	3-4	3-4	3-4	3-4

A. Evaluation

We assess the performance of TRACE based on varying numbers of cables and foreground objects. The evaluation follows a tiered framework introduced in MANIP [1] that is described in Table I and shown visually in Figure 6.

1) Computation and Execution Time: The computation time per TRACE iteration is measured by the total time spent running the HANDLOOM 1.0 tracer, which scales with the number of cables in the trial. The execution time per iteration is measured by the time it takes to complete the interactive perception primitive. For a Tier 4 trial run on a workstation with an AMD Ryzen 7 7700X CPU and an NVIDIA GeForce RTX 4090 GPU, the average computation time and execution time are 6.4 and 13.4 seconds per iteration, respectively.

2) Multi-Cable Tracing Evaluation: We first evaluate the TRACE framework against HANDLOOM 2.0. We conducted 50 physical experiment trials (without foreground objects) distributed across the tiers, with 13 trials in Tiers 1-3 and 11 trials in Tier 4. For each tier, we report the correctly traced cable length relative to the total known cable length in the scene, which is empirically measured. Figure 7 presents the average percent of cable length correctly traced comparing

TABLE II: Average number of interactive perception primitives used per trial (over 60 trials). Note that cluster dilation is used more as complexity increases.

Primitive Used	**Tier 1**	**Tier 2**	**Tier 3**	**Tier 4**
Bimanual Decluttering	1.19x (42.9%)	1.53x (43.4%)	1.33x (20.6%)	1.27x (19.2%)
Cluster Dilation	0.19x (6.3%)	0.53x (12.1%)	1.00x (15.9%)	2.80x (31.3%)
Divergence Push	1.88x (50.8%)	2.33x (44.4%)	4.60x (63.5%)	3.80x (49.4%)
Total Primitives Used	3.25x (100%)	4.40x (100%)	6.93x (100%)	7.87x (100%)

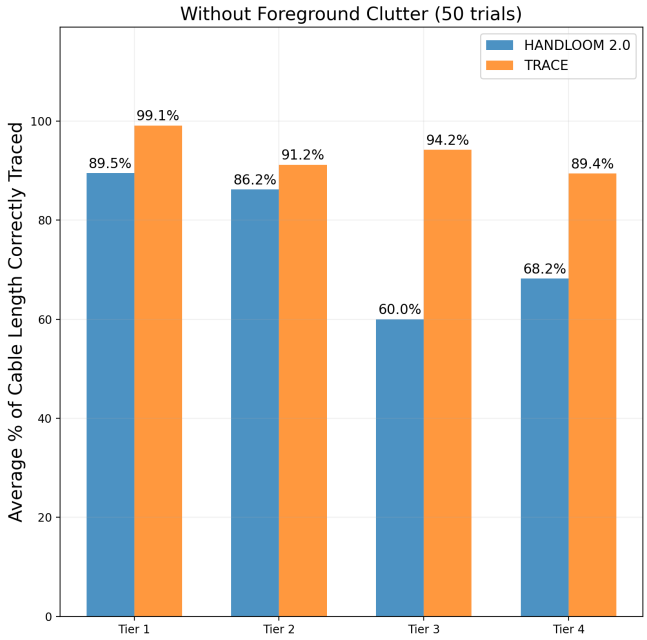

Fig. 7: Without foreground clutter—Average percent of cable length correctly traced comparing TRACE and HANDLOOM 2.0 for 50 physical experiments. Tracing results for HANDLOOM 2.0 were sourced from MANIP [1].

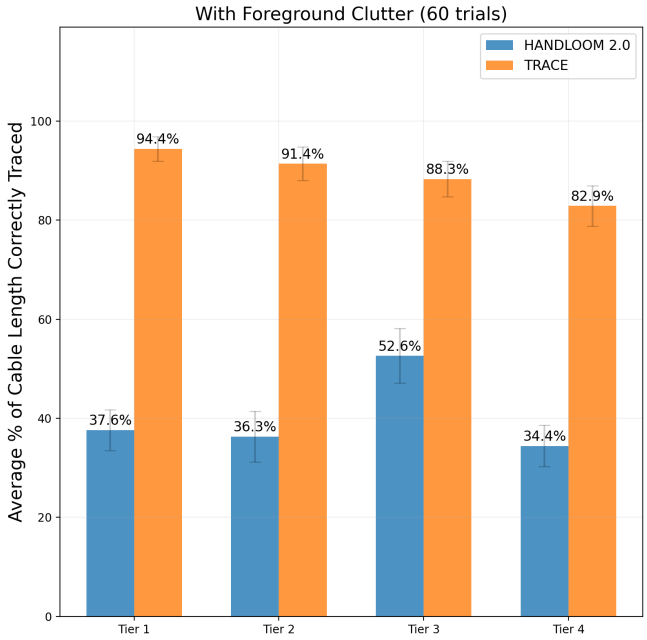

Fig. 8: With foreground clutter—Average percent of cable length correctly traced using TRACE with bimanual decluttering for 60 physical experiments. Vertical bars represent one standard deviation of the mean (15 trials per tier).

both models across different complexity levels. Trace percentages for HANDLOOM 2.0 were sourced from experiments in MANIP [1] (independent from the 50 trials for TRACE). Results suggest that TRACE outperforms HANDLOOM 2.0, achieving comparable correct trace performance in the lower tiers while demonstrating a substantial improvement in more complex tiers, where tracing ambiguities are more prevalent.

3) With Foreground Object Clutter: We conducted 60 physical experiment trials with clutter objects on top of cables, evenly distributed across the 4 tiers. In 32 out of 60 trials, TRACE correctly traced 100% of all cables. Figure 8 presents the average percent of cable length correctly traced with foreground objects. While performance slightly decreases compared to the clutter-free setting, TRACE averages 94.4% in Tier 1 to 82.9% in Tier 4. Additionally, it significantly outperforms HANDLOOM 2.0 on identical scenes by an average of 77.0% improvement across all tiers. Table II illustrates the average number of interactive perception primitives across different tiers. In simpler scenes (Tiers 1–2), Divergence Push is the dominant action, while in more complex configurations (Tiers 3–4), Cluster Dilation becomes more prevalent.

4) Analysis of Failure Modes: To characterize the specific failure modes of TRACE, we manually inspect the results of 60 physical experiments with foreground clutter. We outline these modes in Figure 9. Cases labeled *Success* achieved 100% of cable lengths correctly traced for all cables in the scene, whereas cases labeled *Failure* did not, although all trials achieved a correctly traced cable length of over 60%. We initially classify the failure cases into three modes:

a) *Failed to Identify Divergence Point*: The tracing algorithm could not find any divergence points as candidate sites for an interactive perception primitive. The algorithm completes tracing incorrectly for at least one cable in the scene and matches the connectors of those cable(s) incorrectly as a result, and TRACE terminates.

b) *Ineffective Divergence Push*: The Divergence Push failed to split cables apart and disambiguate the tangential crossing.

c) *Ineffective Cluster Dilation*: The Cluster Dilation failed to open the cluster region enough to disambiguate the cluster.

Within the latter two interactive perception related failure modes, we further classify cases into two modes to identify the root causes:

i) *Cable Springback*: Stiffness caused cables to bounce back to their original position after interactive perception primitive was performed.

ii) *Incorrect Gripper Insertion Point*: The chosen insertion point was too close to a tightly packed cable arrangement, leading to an interactive perception move that failed to reduce cable complexity.

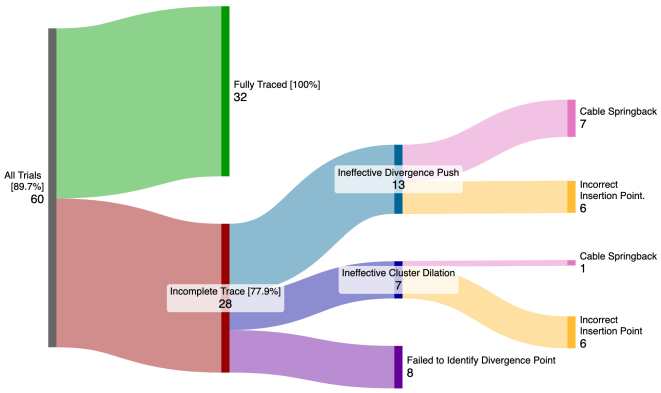

Fig. 9: Sankey diagram detailing failure modes for 60 trials with foreground clutter. The number of cases per mode appears below each label; brackets after leftmost labels denote average percent of cable length correctly traced.

The analysis suggests that the majority of failure modes are caused by the material nature of the cables or cable arrangements that were too dense to effectively disambiguate. This is in line with qualitative observations that increasing the intensity of interactive perception primitives does not significantly improve the effectiveness of the maneuvers.

5) Comparison with Prior Work: We compare TRACE to RT-DLO [6] from 2023, the most comparable tracing system evaluated on cables of similar thickness and color. Although RT-DLO focuses on tracing short cables in the presence of background clutter, it has not been evaluated on three or more longer cables, higher crossing counts, or high-density clusters. In contrast, TRACE assumes that all cables can be segmented from the background. Figure 10 shows a comparison between percent of cable length correctly traced on an image crop evaluated by both systems on a non-cluttered background.

To study these differences and assess the effectiveness of TRACE, we use the divergence points identified by TRACE as regions of ambiguity and select 10 random samples among these across all 4 tiers. For each trial, we locate these divergence points in the "before" image (the raw initial setup) and crop a local 600x600 pixel region around them. We then run RT-DLO on this cropped segment and calculate the average proportion of all cables correctly traced. After TRACE completes all interactive perception primitives, we then run RT-DLO again on the corresponding "after" image, and calculate the final trace percentage. These initial and final results are then compared with those produced by TRACE.

TABLE III: Average percentage of cable length correctly traced in small image crops, comparing TRACE and RT-DLO over 10 trials without background clutter.

Method	Initial	Final	Increase
RT-DLO	57.1%	76.0%	18.9%
TRACE	68.3%	**97.5%**	29.2%
Increase	19.6%	28.3%	

RT-DLO demonstrates a significant increase in accuracy between the initial and final images as shown in Table III, suggesting that TRACE's interactive perception primitives also help RT-DLO perform better. However, the TRACE

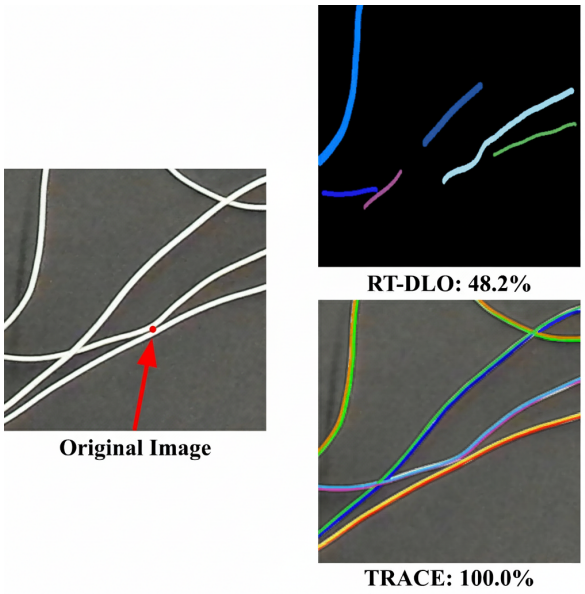

Fig. 10: Comparison between percent of cable length correctly traced on a small image crop, evaluated by RT-DLO (top) and TRACE (bottom) on non-cluttered background. Arrow marks the former divergence point (red).

algorithm significantly outperforms RT-DLO on small crops. We also explored RT-DLO's performance on scenes with long cables. As illustrated in Figure 11, when applied to a full scene with multiple cables, RT-DLO typically traces only a fraction of the total cable length (and often inaccurately).

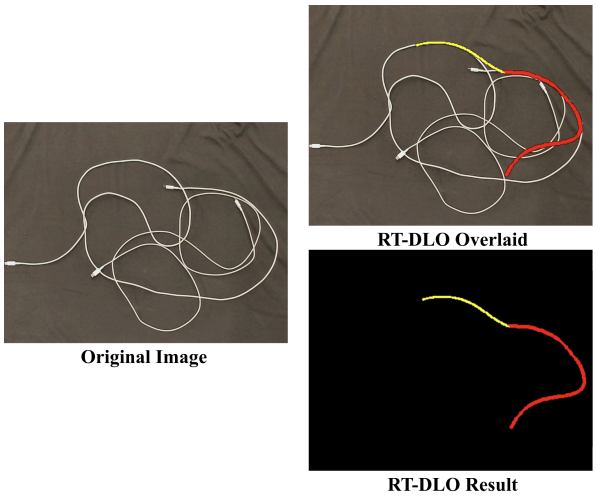

Fig. 11: (Left): Image of a full cable scene on a non-cluttered background. (Top right): Cable length traced by RT-DLO overlaid on the original image. (Bottom right): The isolated RT-DLO trace result.

VI. Limitations

TRACE assumes that all cables are visually distinct against a planar workspace, and that all cable segments are consistently visible from an overhead view. Significant variations in background clutter (especially colors that do not contrast highly against the cables and highly reflective materials) can degrade the tracing algorithm's performance. Figure 12 demonstrates one rare but successful result by TRACE on this type of background.

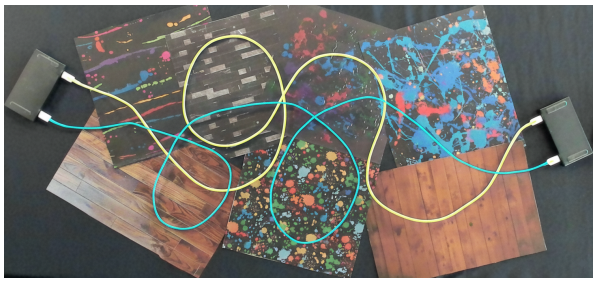

Fig. 12: One successful result by TRACE (100% of cable length correctly traced) on a workspace with highly varied, patterned background clutter.

VII. Future Work

Future work will focus on extending the TRACE framework to handle non-planar scenes and complex, patterned background clutter, moving closer to real-world industrial and domestic environments. Additionally, we will expand the set of interactive perception primitives to resolve ambiguity between a cable and the background itself; for instance, a "lifting" action that may increase contrast between cables and a patterned workspace surface, or a "sliding" motion that may confirm cable continuity across a varied background texture. Future work will also include an adaptive system of interactive perception primitives that recover from failure; for example, if a divergence push fails due to cable springback, subsequent push primitives will vary in intensity to disambiguate the cables more effectively.

References

[1] J. Yu *et al.*, "Manip: A modular architecture for integrating interactive perception for robot manipulation," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2024, pp. 1283–1289.

[2] V. Viswanath *et al.*, "Autonomously Untangling Long Cables," in *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, Jun. 2022.

[3] K. Shivakumar *et al.*, "Sgtm 2.0: Autonomously untangling long cables using interactive perception," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 5837–5843.

[4] V. Viswanath *et al.*, "Handloom: Learned tracing of one-dimensional objects for inspection and manipulation," in *Conference on Robot Learning*, PMLR, 2023, pp. 341–357.

[5] J. Bohg *et al.*, "Interactive perception: Leveraging action in perception and perception in action," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1273–1291, 2017.

[6] A. Caporali, K. Galassi, B. L. Žagar, R. Zanella, G. Palli, and A. C. Knoll, "Rt-dlo: Real-time deformable linear objects instance segmentation," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 11, pp. 11 333–11 342, 2023.

[7] A. Keipour, M. Bandari, and S. Schaal, "Deformable one-dimensional object detection for routing and manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4329–4336, 2022.

[8] P. Kicki, A. Szymko, and K. Walas, "Dloftbs–fast tracking of deformable linear objects with b-splines," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 7104–7110.

[9] X. Huang, D. Chen, Y. Guo, X. Jiang, and Y. Liu, "Untangling multiple deformable linear objects in unknown quantities with complex backgrounds," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 1, pp. 671–683, 2024.

[10] A. Choi, D. Tong, B. Park, D. Terzopoulos, J. Joo, and M. K. Jawed, "Mbest: Realtime deformable linear object detection through minimal bending energy skeleton pixel traversals," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4863–4870, 2023.

[11] W. H. Lui and A. Saxena, "Tangled: Learning to untangle ropes with rgb-d perception," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 837–844.

[12] T. Tang, Y. Fan, H.-C. Lin, and M. Tomizuka, "State estimation for deformable objects by point registration and dynamic simulation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 2427–2433.

[13] J. Xiang and H. Dinkel, "Multidlo: Simultaneous shape tracking of multiple deformable linear objects with global-local topology preservation," in *Representing and Manipulating Deformable Objects at ICRA 2023*, 2023.

[14] J. Xiang *et al.*, "Trackdlo: Tracking deformable linear objects under occlusion with motion coherence," in *IEEE Robotics and Automation Letters*, IEEE, 2023, pp. 6179–6186.

[15] A. Caporali, K. Galassi, R. Zanella, and G. Palli, "Fastdlo: Fast deformable linear objects instance segmentation," *IEEE robotics and automation letters*, vol. 7, no. 4, pp. 9075–9082, 2022.

[16] S. Zhaole, H. Zhou, L. Nanbo, L. Chen, J. Zhu, and R. B. Fisher, "A robust deformable linear object perception pipeline in 3d: From segmentation to reconstruction," in *IEEE Robotics and Automation Letters*, IEEE, 2023, pp. 843–850.

[17] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-supervised learning of state estimation for manipulating deformable linear objects," *IEEE robotics and automation letters*, vol. 5, no. 2, pp. 2372–2379, 2020.

[18] K. Lv, M. Yu, Y. Pu, X. Jiang, G. Huang, and X. Li, "Learning to estimate 3-d states of deformable linear objects from single-frame occluded point clouds," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[20] P. Sundaresan *et al.*, "Untangling Dense Non-Planar Knots by Learning Manipulation Features and Recovery Policies," in *Proceedings of Robotics: Science and Systems*, Virtual, Jul. 2021.

[21] F. Aurenhammer, "Voronoi diagrams—a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, pp. 345–405, 1991.

[22] D. G. Bailey, "An efficient euclidean distance transform," in *International workshop on combinatorial image analysis*, Springer, 2004, pp. 394–408.

[23] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, "Multiscale vessel enhancement filtering," in *Medical Image Computing and Computer-Assisted Intervention — MICCAI'98*, W. M. Wells, A. Colchester, and S. Delp, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 130–137.

[24] I. Jacobi Robotics, *Jacobi motion library – next generation motion planning*, https://docs.jacobirobotics.com, 2024.