LTODO: Learned Tracing of One-Dimensional Objects for Inspection and Manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract: This paper presents LTODO (Learned Tracing of One-Dimensional Objects) a learning-based algorithm that fits traces to an RGB image of one or more cables, hoses, or ropes on a surface and resolves over- vs under-crossings to disambiguate cable state for tasks such as inspection, diagnostics, state-based imitation, and untangling. This work focuses on cables in semi-planar configurations, where crossings involve at most 2 segments. LTODO makes use of neural networks trained with 30,000 simulated examples and 568 real examples to autoregressively estimate traces of cables and classify crossings. Experiments find that in settings with multiple identical cables, LTODO can trace and isolate any cable of interest with 80% accuracy. In single-cable images, LTODO can trace and identify knots with 77% success. When LTODO is incorporated into a bimanual robot system, it enables state-based imitation of knot tying with 80% success, and it successfully untangles 64% of cable configurations across 3 levels of difficulty. Additionally, LTODO demonstrates generalization to knot types and materials (rubber, cloth rope) not present in the training dataset with 85% success. Supplementary material, including an annotated dataset of RGB-D images of knotted cables along with ground-truth traces, can be found at https://sites.google.com/view/cable-tracing.

Keywords: state estimation, deformable manipulation

1 Introduction

In industrial and household settings, unorganized one-dimensional objects such as cables can pose a threat to the safety of workers and at-risk adults by impeding their movement. Moreover, messy or faulty wires can pose a serious risk of entanglement, causing equipment malfunctions and electric shock [1, 2, 3, 4].

To ensure safety, rapid diagnosis and remedial via robot inspection and manipulation of long cables is critical. However, this is constrained by perception challenges. Depth images are prone to noise and estimating cable state from RGB images alone is difficult since cables often fall into complex configurations with many crossings. Long cables can also contain a significant amount of free cable (which we refer to as *slack*), which can occlude and inhibit the perception of knots and crossings from an overhead image. Successful manipulation of long cables requires the ability to trace a cable path in complex configurations and handle the wide distribution of states long cables can fall into.

Prior cable manipulation work bypasses state estimation with object detection and keypoint selection networks for task-specific points based on geometric patterns [5, 6, 7, 8, 9, 10]. Sundaresan et al. [11] employ dense descriptors for cable state estimation in knot tying but focus on thick short cables with loose overhand knots. In comparison, this work tackles the challenges of thin, long cables in semi-planar configurations (at most 2 cable segments per crossing), which often contain dense arrangements and near-parallel cable segments. Although works including Lui and Saxena [12] and

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

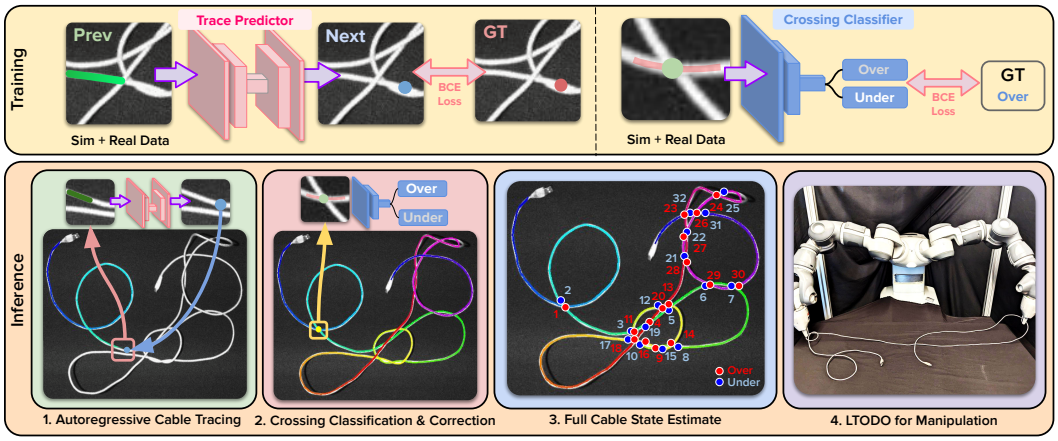

Figure 1: **LTODO:** LTODO consists of two networks. One network (pink) is trained to predict the next point in the trace given a prior context window, and the other network (blue) is trained to classify over- and under-crossings with respect to a cable segment. During inference, LTODO 1) uses the pink network to autoregressively find the most likely trace (shown through a rainbow gradient, from violet to purple, depicting the sequence in which the cable is traced) and 2) performs crossing recognition using the blue network to obtain the full state of the cable (3), where red circles indicate overcrossings and blue circles indicate undercrossings. 4) The state estimate from LTODO can be used for inspection and manipulation.

Huang et al. [13] also perform state estimation of cables, they use analytic methods and do not address the challenging cases mentioned above which we consider. While such analytic methods require many heuristics to score different traces, learning to predict cable traces purely from visual data avoids this problem.

We present Learned Tracing of One-Dimensional Objects (LTODO), shown in Figure 1. This work considers long cables (up to 3 meters in length) in semi-planar configurations, i.e. where each crossing includes at most 2 cable segments. These cable configurations may include knots within a single cable (e.g. overhand, bowline, etc.) or between multiple cables (e.g. carrick bend, sheet bend, etc.). This paper focuses on how learning can facilitate high-accuracy tracing, mostly done analytically previously, and crossing identification to generate a cable state estimate, which we show to be useful for inspection and manipulation tasks. This paper makes the following contributions:

1. LTODO: A cable state estimation algorithm consisting of a learning-based iterative cable tracer and crossing classifier.
2. Novel methods for cable inspection, state-based imitation, knot detection, and autonomous untangling for cables using LTODO.
3. Data from physical robot experiments. Perception results suggest LTODO can correctly trace cables unseen during training with 85% accuracy, trace and segment a single cable in multi-cable settings with 80% accuracy, and detect knots with 77% accuracy. Robot experiments suggest LTODO in a physical system for untangling semi-planar knots achieves 64% untangling success in under 8 minutes and in learning from demonstrations, achieves 80% success.
4. Publicly available code for LTODO and the accompanying data in this work with the data generation code (all included in supplementary material and to be released on publication).

2 Related Work

2.1 Deformable Object Manipulation

This study focuses on tracing long cables, which is crucial for manipulation tasks. Recent advancements in deformable manipulation include cable untangling algorithms [8, 9, 7, 12], fabric smoothing and folding techniques [14, 15, 16, 17, 18, 19, 20], and object placement in bags [21, 22].

Methods for autonomously manipulating deformable objects range from model-free to model-based, the latter of which estimates the state of the object for subsequent planning.

Model-free approaches include reinforcement or self-supervised learning for fabric smoothing and folding [23, 24, 25, 26] and straightening curved ropes [24], or directly imitating human actions [14, 27]. Research by Grannen et al. [8] and Sundaresan et al. [9] employs learning-based keypoint detection to untangle isolated knots without state estimation. Viswanath et al. [5] and Shivakumar et al. [6] extend this approach to long cables (3m) with a learned knot detection pipeline. However, scaling to arbitrary knot types requires impractical amounts of human labels across many conceivable knot types. Our study focuses on state estimation to address this limitation.

Model-based methods for deformable objects employ methods to estimate the state as well as dynamics. Work on cable manipulation includes that by Sundaresan et al. [11], who use dense descriptors for goal-conditioned manipulation. Descriptors have also been applied to fabric smoothing [16], as well as visual dynamics models for non-knotted cables [28, 29] and fabric [17, 28, 30]. Other approaches include learning visual models for manipulation [31], iterative refinement of dynamic actions [32], and using approximate state dynamics with a learned error function [33]. Fusing point clouds across time has shown success in tracking segments of cable provided they are not tangled on themselves [34, 35]. Works from Lui and Saxena [12] and Huang et al. [13] estimate splines of cables before untangling them, and we discuss these methods among others below.

2.2 Tracing One-Dimensional Objects

Prior tracing methods for 1D objects primarily employ analytic approaches [13, 12, 36]. Other works, Jackson et al. [37] and Padoy and Hager [38], optimize splines to trace surgical strings. In contrast, this work focuses on longer cables with a greater variety of configurations, for which analytical methods struggle to differentiate nearby, twisted cables. Some prior work attempts to approach this problem [39] but does not fully estimate cable state, only identifying crossings via analytic methods.

Several prior works approach sub-parts of the problem using learning. Huang et al. [13] focuses on classifying crossings for thick cables, simplifying the task. Song et al. [10] predicts entire traces for short cables as gradient maps but lacks quantitative results for cables and evaluation of long cables with tight knots. Yan et al. [40] refines splines with a self-supervised objective. However, none of these prior works approach tracing autoregressively as we do or demonstrate comparable performance on challenging knots.

3 Problem Statement

Workspace and Assumptions: The workspace is defined by an (x, y, z) coordinate system and contains a fixed overhead RGB-D camera facing the surface and outputs grayscale images. Depth images are used only for manipulation. The RGB image includes 1) only cables (no obstacles), 2) each cable is visually distinguishable from the background, 3) each cable is in a configurations with at least one endpoint visible, and 4) is semi-planar, meaning each crossing contains at most 2 cable segments. We define each cable state for $i = 1, ..., l$ cables to be $\theta_i(s) = \{(x(s), y(s), z(s))\}$ where s is an arc-length parameter that ranges $[0, 1]$, representing the normalized length of the cable. Here, $(x(s), y(s), z(s))$ is the location of a cable point at a normalized arc length of s from the cable's first endpoint. We also define the range of $\theta(s)$—that is, the set of all points on a cable at time t—to be $\mathcal{C}_t$. We assume the cables are visually distinguishable from the background and the background is monochrome. Additional assumptions we make for the manipulation tasks are stated in Section 5.4.2.

Objective: Overall, the objective of LTODO is to estimate the cable state of a cable projected in 2D, which involves generating the pixel-space trace as a function of s of a cable given an overhead grayscale image and resolve whether each of its crossings is "over" or "under". We define a trace as the most probable sequence of spaced pixels that the cable traverses. This state estimate can facilitate manipulation tasks such as those defined in Sections 5.4.2 and 5.3.

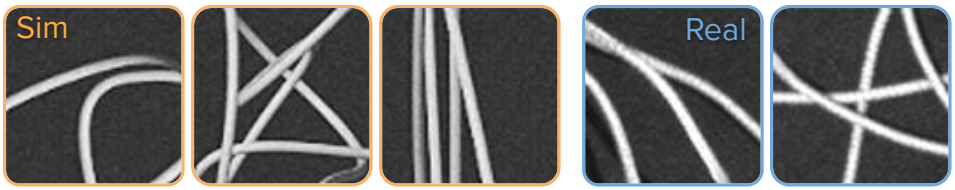

Figure 2: **Simulated and real cropped images of cables used for training LTODO**: On the left are simulated cable crops augmented with Gaussian noise, brightness, and sharpening to match the real images (right).

4 Methods

We present LTODO (Learned Tracing of One-Dimensional Objects), a system that reconstructs cable configurations from grayscale images (Figure 1). It contains a learned cable tracer that estimates the cable's path through the image, which is an autoregressive method that uses predicted points to build a trace, included as an input to the model for the next step of inference. It also comprises a crossing classifier with a correction method that refines predictions, collectively estimating the cable's state.

4.1 Learned Cable Tracer Model

We break the problem of cable tracing down into steps, where each step aims to generate a probability distribution for the next point based on prior points. To achieve this, we employ a learned model that takes an image crop and trace points from previous iterations, predicting a heatmap representing the probability distribution of the next pixel's location. By operating on local information within crops, the model mitigates overfitting and facilitates sim-to-real transfer, focusing on local characteristics rather than global visual and geometric attributes like knots.

More formally, representing the grayscale image as I, and each trace $s_{i,\text{tot}}$ in the image as a sequence of pixels $s_{i,0}, s_{i,1}, ...s_{i,n}$, we break the probability distribution over traces conditioned on the image into smaller, tractable pieces using the chain rule of probability, where f_θ represents a learned neural network and $\text{crop}(I, p)$ represents a crop of image I centered at pixel p, and k represents our context length.

$$P(s_{i,\text{tot}}|\text{s}_{\text{i},0}, \text{I}) = \prod_{j=1}^{n} P(s_{i,j}|s_{i,0}...s_{i,j-1}, I) \approx \prod_{j=1}^{n} f_\theta(s_{i,j}|s_{i,j-k}...s_{i,j-1}, \text{crop}(I, s_{i,j-1}))$$

4.1.1 Dataset and Model Training

We simulate a diverse range of crossing configurations to generate a dataset. We use Blender [41] to create 30,000 simulated grayscale images that closely resemble real observations (Fig. 2). Cable configurations are produced by employing three methods with random Bezier curves: (1) selecting points outside a small exclusion radius around the current point, (2) intentionally creating near-parallel segments, and (3) constraining specific cable segments to achieve a dense and knot-like appearance in certain spatial regions. The curves are colored white and have slightly randomized thicknesses, centered around the thickness of a real cable electrical cable.

We randomly sample image crops along the cable of interest, with a focus on cable crossings (representing 95% of samples). The simulated images are augmented with pixel-wise Gaussian noise with standard deviation of 6, brightness with standard deviation of 5, and sharpening to imitate the appearance of real cables. Additionally, we include a smaller dataset of 568 hand-labeled real cable crop images, sampled such that it comprises approximately 20% of the training examples. Training employs the Adam optimizer [42] with pixelwise binary cross-entropy loss, using a batch size of 64 and a learning rate of 10^{-5}.

4.1.2 Model Architecture and Inference

We use the UNet architecture [43]. We choose trace points spaced approximately 12 pixels apart, chosen by grid search, balancing between adding context and reducing overfitting. We further bal-

ance context and overfitting by tuning the crop size (64×64) and the number of previous points fed into the model (3) through grid search.

Naively training the model for cable tracing in all possible initial directions leads to poor performance, as it would require learning rotational equivariance from data, reducing data efficiency. To overcome this, we pre-rotate the input image, aligning the last two trace points horizontally and ensuring the trace always moves left to right, optimizing the model's capacity for predicting the next trace point, which is a rotationally equivariant task. We then rotate the output heatmap back to the original orientation.

During inference, initializing the trace requires an input of a single start pixel along the cable (in practice, one endpoint). We use an analytic tracer as in [6] to trace approximately 4 trace points and use these points to initialize the learned tracer, which requires multiple previous trace points to predict the next point along the trace.

The tracer autoregressively applies the learned model to extend the trace. The network receives a cropped overhead image centered on the last predicted trace point (64×64 pixels). Previous trace points are fused into a gradient line (shown in Figure 1), forming one channel of the input image. The other two channels contain an identical grayscale image. The model outputs a heatmap ($64 \times 64 \times 1$) indicating the likelihood of each pixel being the next trace point. We greedily select the argmax of this heatmap as the next point in the trace. This process continues until leaving the visible workspace or reaching an endpoint, which is known using similar learned detectors as Shivakumar et al. [6].

4.2 Over/Undercrossing Predictor

4.2.1 Data and Model Input

We use 20×20 simulated and real over/undercrossing crops. The 568 real images are oversampled such that they are seen 20% of the time during training. Augmentation methods applied to the cable tracer are also used here to mimic the appearance of real cable crops. The network receives a $20 \times 20 \times 3$ input crop. The first channel encodes trace points fused into a line, representing the segment of interest. To exploit rotational invariance, the crop is rotated to ensure the segment's first and last points are horizontal. The second channel consists of a Gaussian heatmap centered at the target crossing position, providing positional information to handle dense configurations. The third channel encodes the grayscale image of the crop.

4.2.2 Model Architecture and Inference

We use a ResNet-34 classification model with a sigmoid activation to predict scores between 0 and 1. The model is trained using binary cross-entropy loss. We determine the binary classification using a threshold of 0.275 (explanation for this number is in the appendix Section 7.2.1). The algorithm uses cable semi-planarity to adjusts probabilities of crossings based on the over/undercrossing classifier's predictions, favoring the higher confidence detection and updating the probability of the crossing to ($1-$ the original value), storing the confidences for subsequent tasks.

The learned cable tracer, over/undercrossing predictor, and crossing correction method combined together result in a full cable state estimator: LTODO.

4.3 Using LTODO in Downstream Applications

Tracing in Multi-cable Settings: We apply LTODO to inspection in multi-cable settings for applications like locating the power adapter of a cable tangled with other visually similar cables given the endpoint. LTODO is fed an endpoint of interest and returns a trace to the adapter connected to it.

Learning from Demonstrations: We apply LTODO to enable physical, state-based, one-shot imitation of knot tying despite the presence of distractor cables, which is intractable for a policy operating on RGB observations rather than underlying state. Demonstrations are pick-place actions, parameterized by absolute arc-length or arc-length relative to crossings. More details are in Appendix Section 7.4.

Table 1: Generalization of LTODO to Different Cable Types

Cable Reference	Length (m)	Color	Texture	Physical Properties	LTODO Succ. Rate
TR (trained with)	2.74	White/gray	Braided	Slightly stiff	6/8
1	2.09	Gray	Rubbery	Slightly thicker than TR	7/8
2	4.68	Yellow with black text	Rubbery and plastic	Very stiff	8/8
3	2.08	Tan	Rubbery	Highly elastic	7/8
4	1.79	Bright red	Braided	Flimsy	6/8
5	4.61	White	Braided	Flimsy	6/8

Robot Cable Untangling: In the cable untangling application, LTODO is combined with analytic knot detection, untangling point detection techniques, and bi-manual robot manipulation primitives to create a system for robot untangling. This is expanded on in the Appendix in Section 7.3.

5 Physical Experiments

We evaluate LTODO on 1) tracing cables unseen during training, 2) cable inspection in multi-cable settings, 3) learning knot tying from demonstrations, 4) knot detection, and 5) untangling.

The workspace has a 1 m $\times$ 0.75 m surface, a bimanual ABB YuMi robot, and an overhead Photoneo PhoXi camera with $773 \times 1032 \times 4$ RGB-D observations. Although there are 3 color channels, images outputted by the PhoXi are grayscale. The workspace is padded with a 5cm-tall foam and covered with black cloth. See Appendix, Section 7.1 for details on failure modes for each experiment.

5.1 Using LTODO for Tracing Cables Unseen During Training

For this perception experiment, the workspace contains a single cable with one of the following knots: overhand, figure-eight, overhand honda, or bowline. Here, we provide LTODO with the two endpoints to test the tracer in isolation, independent of endpoint detection. We report a success if the progression of the trace correctly follows the path of the cable without deviating.

Results in Table 1 show LTODO can generalize to cables with varying appearances, textures, lengths, and physical properties. Cables can also be seen in Appendix Figure 5. LTODO performs comparably on cable TR (the cable it was trained with) as it does on any of the other cables.

5.2 Using LTODO for Cable Inspection in Multi-Cable Settings

Figure 3: **Multi-cable tracing**: **Top row:** illustrative examples of each of the 3 tiers of difficulty for multi-cable tracing experiments. **Bottom row:** Corresponding successful traces outputted by the learned tracer.

For cable inspection, the workspace contains a power strip. Attached to the power strip are three MacBook adapters with two 3 m USB C-to-C cables and one 2 m plain white USB-C to MagSafe 3 cable (shown in Figure 3). The goal is to provide the trace of a cable and identify the relevant adaptor given the endpoint. We evaluate perception in multi-cable settings across 3 tiers of difficulty.

1. **Tier A1**: No knots; cables are dropped onto the workspace, one at a time.
2. **Tier A2**: Each cable is tied with a single knot (figure-eight, overhand, overhand honda, bowline, linked overhand, or figure-eight honda) measuring 5-10 cm in diameter, and subsequently dropped randomly onto the workspace one by one.
3. **Tier A3**: Similar to tier A2 but contains the following 2-cable knots (square, carrick bend, and sheet bend) with up to three knots in the scene.

Across all 3 tiers, we assume the cable of interest cannot exit and re-enter the workspace and that crossings must be semi-planar. Additionally, we pass in the locations of all three adapters to the

tracer and an endpoint to start tracing from. To account for noise in the input images, we take 3 images of each configuration. We count the experiment as a success if a majority (2 of the 3 images) have the correct trace (reaching their corresponding adapter); otherwise, we report a failure.

We compare the performance of the learned tracer from LTODO against an analytic tracer from Shivakumar et al. [6] as a baseline, using scoring rules inspired by Lui and Saxena [12] and Keipour et al. [36]. The analytic tracer explores potential paths and selects the most likely trace based on a scoring metric [6], prioritizing paths that reach an endpoint, have minimal angle deviations, and have high coverage scores. Table 2 shows that the learned tracer significantly outperforms the baseline analytic tracer on all 3 tiers of difficulty with a total of 80% success across the tiers.

5.3 Using LTODO for Physical Robot Knot Tying from Demonstrations

Knot-tying demonstrations are conducted on a nylon rope of length 147 cm and a diameter of 7 mm. We tune our demonstrations to work on plain backgrounds; however, during rollouts, we add distractor cables that intersect the cable to be tied (cable of interest). These distractor cables are of identical types to the cable of interest; thus, manipulating the correct points requires accurate cable state estimation from LTODO. Although these cables are thicker, more twisted, and less stiff than the cable LTODO is trained with, LTODO is able to generalize, tracing the cable successfully in 13 out of 15 cases. We evaluate the policy executed by the YuMi bimanual robot on the following: **Tier B1:** 0 distractors, **Tier B2:** 1 distractor, and **Tier B3:** 2 distractors. We consider the execution a success when a knot has been tied (i.e. lifting the endpoints results in a knot's presence).

Results in Table 4 show 80% robotic knot tying success and 86% tracing success across the 3 tiers, suggesting LTODO can apply policies learned from demonstrations in the real world, even in the presence of distractors.

5.4 Using LTODO for Knot Detection and Physical Robot Untangling

To test LTODO applied to the knot detection task and physical untangling, we use a single 3 m white, braided USB-A to micro-USB cable.

5.4.1 Knot Detection

The details on the state-based knot detection method, which uses the sequence of over- and under-crossings to identify knots, can be found in the Appendix Section 7.3.2.

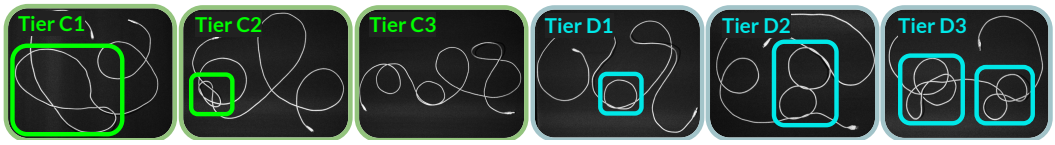

Figure 4: **Starting configurations** for the tiers for LTODO experiments and robot untangling experiments.

We evaluate LTODO on 3 tiers of cable configurations, shown in Figure 4. The ordering of the categories for these experiments does not indicate varying difficulty. Rather, they are 3 categories of knot configurations to test LTODO on.

1. **Tier C1**: Loose (35-40 cm in diameter) figure-eight, overhand, overhand honda, bowline, linked overhand, and figure-eight honda knots.
2. **Tier C2**: Dense (5-10 cm in diameter) figure-eight, overhand, overhand honda, bowline, linked overhand, and figure-eight honda knots.
3. **Tier C3**: Fake knots (trivial configurations positioned to appear knot-like from afar).

We evaluate LTODO and the following 3 baselines on the 3 tiers.

1. SGTM 2.0 [6] perception system: using a Mask R-CNN model trained on overhand and figure-eight knots for knot detection.
2. LTODO (-LT): replacing the Learned Tracer with the same analytic tracer from Shivakumar et al. [6] as described in Section 5.2 combined with the crossing identification.

Table 2: Multi-Cable Tracing

Tier	Analytic	Learned
A1	3/30	**26/30**
A2	2/30	**23/30**
A3	1/30	**23/30**

Table 3: Knot Detection Experiments

Tier	SGTM 2.0	LTODO (-LT)	LTODO (-CC)	LTODO
C1	2/30	14/30	20/30	**24/30**
C2	**28/30**	8/30	21/30	26/30
C3	12/30	14/30	0/30	**19/30**

Table 4: Learning From Demos

Tier	Succ.	Corr. Trace
B1	5/5	5/5
B2	4/5	4/5
B3	3/5	4/5

Table 5: Robot Untangling Experiments (90 total trials)

	Tier D1		**Tier D2**		**Tier D3**	
	SGTM 2.0	LTODO	SGTM 2.0	LTODO	SGTM 2.0	LTODO
Knot 1 Succ.	11/15	**12/15**	6/15	**11/15**	9/15	**14/15**
Knot 2 Succ.	-	-	-	-	2/15	**6/15**
Verif. Rate	**11/11**	8/12	**6/6**	6/11	**1/2**	2/6
Knot 1 Time (min)	1.1±0.1	2.1±0.3	3.5±0.7	3.9±1.1	1.8±0.4	2.0±0.4
Knot 2 Time (min)	-	-	-	-	3.1±1.2	7.5±1.6
Verif. Time (min)	5.7±0.9	6.1±1.4	6.4±1.8	10.1±0.7	5.4	9.6±1.5

3. LTODO (-CC): using the learned tracer and crossing identification scheme to do knot detection without Crossing Cancellation, covered in the appendix (Section 7.3.3).

We report the success rate of each of these algorithms as follows: if a knot is present, the algorithm is successful if it correctly detects the first knot (i.e. labels its first undercrossing); if there are no knots, the algorithm is successful if it correctly detects that there are no knots.

As summarized in Table 3, knot detection using LTODO considerably outperforms SGTM 2.0, LTODO (-LT), and LTODO (-CC) on tiers C1 and C3. SGTM 2.0 marginally outperforms LTODO in tier C2. This is because SGTM 2.0's Mask R-CNN is trained on dense overhand and figure-eight knots, which are visually similar to the knots in tier C2.

5.4.2 Physical Robot Untangling

For the untangling system based on LTODO, we compare performance against SGTM 2.0 [6], the current state-of-the-art algorithm for untangling long cables, using the same 15-minute timeout on each rollout and the same metrics for comparison. We evaluate LTODO deployed on the ABB YuMi bimanual robot in untangling performance on the following 3 levels of difficulty (Figure 4), where all knots are upward of 10 cm in diameter:

Tier D1: Cable with overhand, figure-eight, or overhand honda knot; total crossings ≤ 6.

Tier D2: Cable with bowline, linked overhand, or figure-eight honda knot; total crossings $\in [6, 10)$.

Tier D3: Cable with 2 knots (1 each from tiers D1 and D2); total crossings $\in [10, 15)$.

Table 5 shows that our LTODO-based untangling system achieves a higher untangling success rate (29/45) than SGTM 2.0 (19/45) across 3 tiers of difficulty, although SGTM 2.0 is faster. The slower speed of LTODO is attributed to the requirement of a full cable trace for knot detection which is inhibited by the fact that the cable is $3\times$ as long as the width of the workspace, leading to additional reveal moves before performing an action or verifying termination. On the other hand, SGTM 2.0 does not account for the cable exiting the workspace, benefiting speed, but failing to detect off-workspace knots, leading to premature endings of rollouts without fully untangling the cable.

6 Limitations and Future Work

LTODO assumes cables are not occluded by large, non-cable objects, are in semi-planar configurations, and are distinguishable from the background, which is assumed to be plain. Additionally, LTODO is not trained on cables with a wide range of physical (e.g. thickness) and material (elasticity, bending radius) properties, such as loose string or rubbery tubing; and certain empirical experimentation suggests that for significantly thicker ropes and threads with the ability form "kinks", performance of LTODO suffers due to the out-of-distribution nature of these cases. Future work will aim to mitigate these problems and also generalize across more broadly varying backgrounds and properties of cable, perhaps via broader datasets. Another area of work will involve sampling multiple likely traces from the model and using resultant uncertainty as a signal downstream as a means of mitigating noise. Lastly, LTODO requires an initialization of the trace (provided by an analytic tracer); future work will aim to train the learned tracer to begin from the endpoint itself.

References

[1] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research*, 37(7):688–716, 2018.

[2] H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, and J. Schmidhuber. A system for robotic heart surgery that learns to tie knots using recurrent neural networks. *Advanced Robotics*, 22 (13-14):1521–1537, 2008.

[3] J. Van Den Berg, S. Miller, D. Duckworth, H. Hu, A. Wan, X.-Y. Fu, K. Goldberg, and P. Abbeel. Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations. In *2010 IEEE International Conference on Robotics and Automation*, pages 2074–2081. IEEE, 2010.

[4] Y. Yamakawa, A. Namiki, M. Ishikawa, and M. Shimojo. One-handed knotting of a flexible rope with a high-speed multifingered hand having tactile sensors. In *2007 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 703–708. IEEE, 2007.

[5] V. Viswanath, K. Shivakumar, J. Kerr, B. Thananjeyan, E. Novoseller, J. Ichnowski, A. Escontrela, M. Laskey, J. E. Gonzalez, and K. Goldberg. Autonomously untangling long cables. *Robotics: Science and Systems (RSS)*, 2022.

[6] K. Shivakumar, V. Viswanath, A. Gu, Y. Avigal, J. Kerr, J. Ichnowski, R. Cheng, T. Kollar, and K. Goldberg. Sgtm 2.0: Autonomously untangling long cables using interactive perception. *arXiv preprint arXiv:2209.13706*, 2022.

[7] V. Viswanath, J. Grannen, P. Sundaresan, B. Thananjeyan, A. Balakrishna, E. Novoseller, J. Ichnowski, M. Laskey, J. E. Gonzalez, and K. Goldberg. Disentangling dense multi-cable knots. *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[8] J. Grannen, P. Sundaresan, B. Thananjeyan, J. Ichnowski, A. Balakrishna, M. Hwang, V. Viswanath, M. Laskey, J. E. Gonzalez, and K. Goldberg. Untangling dense knots by learning task-relevant keypoints. *Conference on Robot Learning*, 2020.

[9] P. Sundaresan, J. Grannen, B. Thananjeyan, A. Balakrishna, J. Ichnowski, E. Novoseller, M. Hwang, M. Laskey, J. E. Gonzalez, and K. Goldberg. Untangling dense non-planar knots by learning manipulation features and recovery policies. *Proc. Robotics: Science and Systems (RSS)*, 2021.

[10] Y. Song, K. Yang, X. Jiang, and Y. Liu. Vision based topological state recognition for deformable linear object untangling conducted in unknown background. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 790–795, 2019. doi:10.1109/ROBIO49542.2019.8961652.

[11] P. Sundaresan, J. Grannen, B. Thananjeyan, A. Balakrishna, M. Laskey, K. Stone, J. E. Gonzalez, and K. Goldberg. Learning rope manipulation policies using dense object descriptors trained on synthetic depth data. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2020.

[12] W. H. Lui and A. Saxena. Tangled: Learning to untangle ropes with RGB-D perception. In *2013 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 837–844. IEEE, 2013.

[13] 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*, pages 1–13, 2023. doi:10.1109/TASE.2023.3233949.

[14] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali, et al. Deep imitation learning of sequential fabric smoothing from an algorithmic supervisor. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[15] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal, and D. Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. In *Conference on Robot Learning*, pages 192–202. PMLR, 2022.

[16] A. Ganapathi, P. Sundaresan, B. Thananjeyan, A. Balakrishna, D. Seita, J. Grannen, M. Hwang, R. Hoque, J. E. Gonzalez, N. Jamali, et al. Learning to smooth and fold real fabric using dense object descriptors trained on synthetic color images. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2021.

[17] R. Hoque, D. Seita, A. Balakrishna, A. Ganapathi, A. K. Tanwani, N. Jamali, K. Yamane, S. Iba, and K. Goldberg. Visuospatial foresight for multi-step, multi-task fabric manipulation. In *Proc. Robotics: Science and Systems (RSS)*, 2020.

[18] T. Kollar, M. Laskey, K. Stone, B. Thananjeyan, and M. Tjersland. Simnet: Enabling robust unknown object manipulation from pure synthetic data via stereo. In *Conference on Robot Learning*, pages 938–948. PMLR, 2022.

[19] B. Thananjeyan, J. Kerr, H. Huang, J. E. Gonzalez, and K. Goldberg. All you need is luv: Unsupervised collection of labeled images using invisible uv fluorescent indicators, 2022. URL https://arxiv.org/abs/2203.04566.

[20] R. Hoque, K. Shivakumar, S. Aeron, G. Deza, A. Ganapathi, A. Wong, J. Lee, A. Zeng, V. Vanhoucke, and K. Goldberg. Learning to fold real garments with one arm: A case study in cloud-based robotics research. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 251–257, 2022. doi:10.1109/IROS47612.2022.9981253.

[21] D. Seita, P. Florence, J. Tompson, E. Coumans, V. Sindhwani, K. Goldberg, and A. Zeng. Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2021.

[22] L. Y. Chen, B. Shi, D. Seita, R. Cheng, T. Kollar, D. Held, and K. Goldberg. Autobag: Learning to open plastic bags and insert objects, 2022. URL https://arxiv.org/abs/2210.17217.

[23] J. Matas, S. James, and A. J. Davison. Sim-to-real reinforcement learning for deformable object manipulation. In *Conf. on Robot Learning (CoRL)*, 2018.

[24] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel. Learning to manipulate deformable objects without demonstrations. *arXiv preprint arXiv:1910.13439*, 2019.

[25] R. Lee, D. Ward, A. Cosgun, V. Dasagi, P. Corke, and J. Leitner. Learning arbitrary-goal fabric folding with one hour of real robot experience. In *Conf. on Robot Learning (CoRL)*, 2020.

[26] Y. Avigal, L. Berscheid, T. Asfour, T. Kröger, and K. Goldberg. Speedfolding: Learning efficient bimanual folding of garments, 2022. URL https://arxiv.org/abs/2208.10552.

[27] J. Schulman, J. Ho, C. Lee, and P. Abbeel. Learning from demonstrations through the use of non-rigid registration. In *International Symposium of Robotics Research*, 2013.

[28] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto. Learning predictive representations for deformable objects using contrastive estimation. In *Conf. on Robot Learning (CoRL)*, 2020.

[29] A. Wang, T. Kurutach, K. Liu, P. Abbeel, and A. Tamar. Learning robotic manipulation through visual planning and acting. *Robotics: Science and Systems (RSS)*, 2019.

[30] X. Lin, Y. Wang, Z. Huang, and D. Held. Learning visible connectivity dynamics for cloth smoothing. In *Conference on Robot Learning*, pages 256–266. PMLR, 2022.

[31] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. *CoRR*, abs/1703.02018, 2017. URL http://arxiv.org/abs/1703.02018.

[32] C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Iterative residual policy for goal-conditioned dynamic manipulation of deformable objects. In *Proceedings of Robotics: Science and Systems (RSS)*, 2022.

[33] D. McConachie, T. Power, P. Mitrano, and D. Berenson. Learning when to trust a dynamics model for planning in reduced state spaces. *CoRR*, abs/2001.11051, 2020. URL https://arxiv.org/abs/2001.11051.

[34] J. Schulman, A. Lee, J. Ho, and P. Abbeel. Tracking deformable objects with point clouds. In *2013 IEEE International Conference on Robotics and Automation*, pages 1130–1137, 2013. doi:10.1109/ICRA.2013.6630714.

[35] T. Tang and M. Tomizuka. Track deformable objects from point clouds with structure preserved registration. *The International Journal of Robotics Research*, 41(6):599–614, 2022. doi:10.1177/0278364919841431. URL https://doi.org/10.1177/0278364919841431.

[36] A. Keipour, M. Bandari, and S. Schaal. Deformable one-dimensional object detection for routing and manipulation. *CoRR*, abs/2201.06775, 2022. URL https://arxiv.org/abs/2201.06775.

[37] R. C. Jackson, R. Yuan, D.-L. Chow, W. S. Newman, and M. C. Çavuşoğlu. Real-time visual tracking of dynamic surgical suture threads. *IEEE Transactions on Automation Science and Engineering*, 15(3):1078–1090, 2018. doi:10.1109/TASE.2017.2726689.

[38] N. Padoy and G. Hager. Deformable tracking of textured curvilinear objects. In *Proceedings of the British Machine Vision Conference*, pages 5.1–5.11. BMVA Press, 2012. ISBN 1-901725-46-4. doi:http://dx.doi.org/10.5244/C.26.5.

[39] P. Parmar. Use of computer vision to detect tangles in tangled objects. In *2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013)*. IEEE, dec 2013. doi:10.1109/iciip.2013.6707551.

[40] 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*, 5(2):2372–2379, 2020. doi:10.1109/LRA.2020.2969931.

[41] B. O. Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL http://www.blender.org.

[42] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[43] P. Iakubovskii. Segmentation models pytorch. https://github.com/qubvel/segmentation_models.pytorch, 2019.

[44] K. Reidemeister. *Knot theory*. BCS Associates, 1983.