Real-time State Estimation of Deformable Objects with Dynamical Simulation

Shiyu Jin*,1, Changhao Wang*,1, Xinghao Zhu*,1, Te Tang[2] and Masayoshi Tomizuka[1]

***Abstract*— Estimating the state of deformable objects is vital for manipulation, while it is also challenging due to high degrees of freedom and the nonlinearity of the dynamics model. In order to achieve robust state estimation, we propose a novel framework shown in Fig. 1, which includes point cloud recovery, node registration, feedback linearization controller, and dynamical simulation modules. Compared with previous works, the point cloud recovery step is able to robustly provide a complete point cloud of the object even under massive occlusions. In addition, the feedback linearization controller is able to stabilize the rope tracking procedure via applying a control law that first cancels higher-order terms in the dynamic equation and then uses an additional PD control law to control the remaining linear dynamics. Experimental results validate the effectiveness and robustness of the proposed framework. The experimental videos can be found at [1].**

I. INTRODUCTION

Manipulating deformable objects has great values for a wide variety of robot applications such as food delivery, medical surgery, cable assembly, and household works. In order to achieve robust performances for those challenging manipulation tasks, estimating the state of deformable objects is crucial. In this work, we focus on state estimation of linear deformable objects such as ropes and cables.

Researchers have studied the rope manipulation tasks [2]–[11] and suggested two main challenges. First, there is no compact state representation of non-rigid objects. Many control and planning algorithms need clear states in order to optimize the cost function. The lack of canonical form often limits the usage of those approaches. Furthermore, perception challenges, due to occlusions, make the state estimation even harder. Second, the dynamics of non-rigid objects make planning tough. Due to high degrees of freedom of the rope, motions of non-rigid objects are unpredictable, which makes it challenging to approximate the highly non-linear dynamics functions. Algorithms like linear-quadratic regulator and model predictive control heavily rely on the dynamical functions.

In recent years, several studies have proposed to tackle the challenges in state estimation and dynamics approximation of deformable objects. Learning-based methods have achieved promising results on rope manipulation tasks, but require a large amount of training data in order to learn rope's state representation and the dynamics transfer function. We circumvent these problems by dividing rope manipulation into rope tracking and rope motion planning. To be more specific, we utilize a simulated rope to track its real counterpart and plan motions based on the simulated dynamics. Compared with learning-based methods, using such an interpretable representation not only alleviates the difficulty in approximating rope dynamics but also makes the representation task-agnostic. In this article, we primarily focus on rope tracking and state estimation.

One class of methods that focuses on the challenges in state estimation is learning latent representations from pixels. Some approaches [6], [9] do not enforce the geometrical constraints of the rope in the learned space, which limits the interpretability of the representation. In contrast, [7] imposed contrastive loss in order to bring the positive pairs (similar rope's configurations) closer in the latent space and the negative pairs (dissimilar rope's configurations) further away. By contrastively learning pixel-wise descriptors, [8] claimed that dense depth object descriptors could enforce geometrical information on the learned visual representations of a deformable rope.

Another class of approaches is to track the rope by fitting a Gaussian Mixture Model (GMM). Prior works studied the tracking of deformable objects in the presence of occlusions. [2]–[4] utilized a simulated rope and computed its reference pose by non-rigid registration. [10] claimed the redundancy of the physics engine. From the experiments, however, we found that the physical simulation plays an important role in enforcing geometrical constraints. The probability-based method is statistically correct in point registration, but physically wrong to represent a rope.

In this paper, we introduce a rope tracking approach with three main parts as shown in Fig. 1: point cloud recovery, point registration, and feedback linearization controller in the physical simulation. In the point registration, a rope is represented by N nodes and registered to sensor observations with a structure preserved registration (SPR) algorithm [11]. From experiments, we have noticed the SPR algorithm is insufficient in handling large area occlusions. Thus, we introduce a mask-based point recovery preprocess module to enhance the robustness of tracking. To consider the dynamical constraints, a simulated rope is constructed in the physics engine and controlled by a feedback linearization controller. We show by experiments that the overall framework is able

* Denotes equal contributions.
[1]Mechanical Systems Control Lab, Department of Mechanical Engineering, University of California, Berkeley, CA, USA. {jsy,changhaowang,zhuxh,tomizuka}@berkeley.edu
[2]FANUC Advanced Research Laboratory, FANUC America Corporation, Union City, CA, USA. Te.Tang@fanucamerica.com

Workshop on Robotic Manipulation of Deformable Objects (ROMADO)
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2020

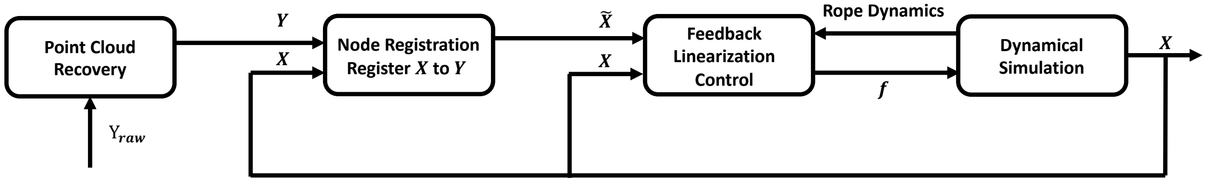

Fig. 1. Overall framework for rope tracking. Point cloud recovery, node registration, feedback linearization controller, and dynamical simulation run in a closed loop to provide a robust tracking result.

to track the rope robustly.

The rest of this paper is organized as follows: Chapter II briefly introduces our previous work; Chapter III talks about the implementation details including point cloud recovery and the feedback linearization controller; Chapter IV presents preliminary experimental results; Chapter V discusses the proposed framework and points out potential future works.

II. Background and Previous Works

State estimation and task planning are two of the main challenges in deformable object manipulation. We have extensively studied both topics.

For state estimation, the difficulties lie in several aspects: 1) lacking explicit correspondence between the object point cloud and the object nodes; 2) sensor noise and outliers; 3) missing points in the object point cloud due to occlusions. To cope with the above challenges, a novel state observer for tracking soft objects was designed in [2]. It utilized a Gaussian mixture model (GMM) to construct the estimator, where each object node is regarded as a Gaussian centroid, and the point cloud is the data set sampled from the Gaussian mixture model. The node position, i.e., the mean of each Gaussian component, can be estimated by maximizing the likelihood of point registrations. To further improve the robustness of estimation under occlusion, a topological regularization was applied in [11] to the mixture model, which preserves the topological structure of the object during registration both locally and globally.

Both imitation learning and optimization-based methods have been studied to address the task planning problem. Inspired by the idea of imitation learning, we developed a 'trajectory warping' algorithm [3], which is able to generalize the human-taught trajectory from a specific environment to a new environment. From the experiments, we observed that the 'trajectory warping' algorithm is not able to maintain the structural information of the deformable objects. To deal with this problem, a tangent space mapping (TSM) algorithm [12], which maps the deformable object in the tangent space instead of the Cartesian space, was developed. The new algorithm was shown to be robust to the changes in the object's pose/shape, and the object's final shape was similar to that of training. In addition to imitation learning-based methods, we also proposed a robust local linear model approximation method [5], where we robustly approximated the model of the object in real-time and then used an optimization-based trajectory planner to find the best movement of the robot.

In this paper, we specifically focus on the state estimation for deformable objects. From the observation that our previous tracking algorithm suffers from massive point cloud occlusion, we develop a point cloud recovery module to deal with this problem for robust tracking performance. Another major contribution of this paper is to study the control law to be applied to the simulated object in the physics engine in order to track the desired position.

III. Algorithm detail

A. Point Cloud Recovery

The state estimator registers the old simulated rope to the point cloud in order to update the simulation. During tracking, human hands or robot arms may occlude the object resulting in a partially observed point cloud. From experiments, we found SPR [11] is robust when the occluded part is small. However, if a large portion of the point cloud is missing (Fig. 2) or the tip of the cable is occluded, SPR would fail to estimate the correct cable state.

Inspired by the background subtraction method in computer vision, we proposed to use a foreground mask to recover the occluded point cloud (Fig. 3). Fig. 3 (a) shows the environment background captured with a stationary RGB-depth camera. (b) and (c) are RGB images with their color-filtered point clouds in time step $t-1$ and t. (d) is the foreground mask constructed by subtracting background from frame t.

The goal of the point cloud recovery is to complete point cloud in frame t using the foreground mask and the recovered point cloud in frame $t-1$. Fig. 4 provides an example of the process. First, point clouds at frame $t-1$ and t are projected to the RGB image plane as (a) and (c) show. Second, we compute a foreground mask in frame t as shown in (b). Third, the projected image in frame $t-1$ is multiplied in pixel with the mask to obtain the complementary part. Finally, the complement is combined with origin point clouds in frame t to get (d). The advantage of this operation is that it could distinguish rope movement from occlusion. For the convenience of discussion, we divide rope's point clouds in

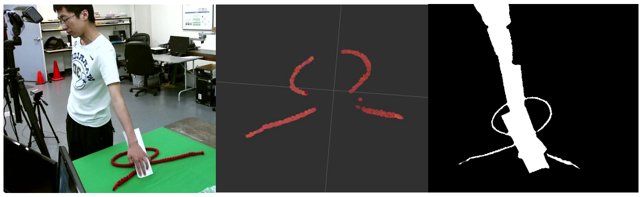

Fig. 2. (left) Kinect is occluded by human arm. (middle) Partial observed point cloud. (right) Foreground mask.

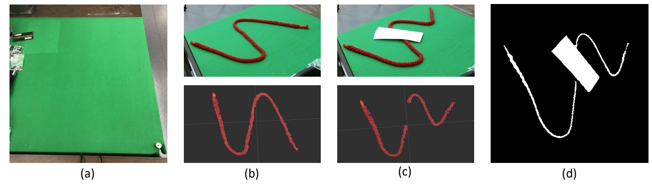

Fig. 3. (a) Background. (b) $t-1$ frame. (c) t frame. (d) Foreground mask.

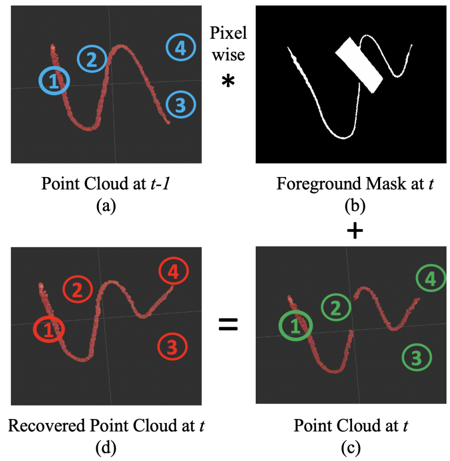

Fig. 4. Point Cloud Recovery

each frame into four parts and labeled 1 to 4 in Fig. 4. The point cloud in the frame t (c) at part 3 is missing compared to the frame $t-1$ (a). However, the foreground mask (b) shows no occlusion at part 3, so the corresponding point cloud at frame $t-1$ will be discarded. The proposed algorithm is summarized in Algorithm 1.

Due to sensor noise and mismatch on RGB and depth cameras, there are alignment errors when constructing the foreground mask. This would result in a segmentation error of the point cloud. In our case, we prefer a false-negative mask rather than a false-positive. In other words, we could discard some occluded points, but not preserve inexistent points. Thus, an erosion operation is applied on the foreground mask to avoid false-positive masks.

Algorithm 1: Point Cloud Recovery

```
Initialize the environment, record the background;
Put the rope in the environment, obtain the point
  cloud of the rope using a color filter, t+ = 1;
while Tracking do
    Obtain the point cloud of the rope using a color
      filter;
    Obtain the foreground mask by subtracting the
      background from the current frame;
    for each pixel of the frame do
        if point cloud in frame t is True then
            preserve the point cloud ;
        else if point cloud in frame t − 1 is True and
          foreground mask is True then
            preserve the point cloud ;
        else
            discard the point cloud ;
        end
    end
    t = t + 1;
end
```

B. Rope Registration and Interpolation

The rope can be represented by N nodes X^t at each time step. X^t are registered from the recovered point cloud Y^t_{raw} using SPR [11]. SPR maximizes the likelihood of a standard GMM with external cost on both the global and local structure of the rope. The registration result, however, is a local optimal. The constraint which compels adjacent node distance: $||x_i - x_{i-1}||^2 \equiv c \geq 0$ is not strictly satisfied. In practice, this soft constraint results in 'zig-zag' shapes of tracking. To handle this problem, we add a cubic spline interpolation step to uniformly resample the node positions along the rope after each registration. Therefore, our solution satisfies both the global and local structure via SPR and also avoids potential 'zig-zag' shapes during tracking.

C. Dynamical Simulation with Physic Engine

As a physical entity, the object needs to satisfy a series of physical laws, such as kinematics, dynamics, and collision constraints, which we do not consider in the node registration step. To account for those constraints, the Bullet physics engine is used to simulate the dynamical behavior of the rope.

Most of the previous works [2], [4] used PD control laws to track the target position. Though Tang et al [2] claimed to use an impedance controller for rope tracking, the control law they used was still a PD. Considering that the dynamics of the rope is nonlinear in the physic engine, a simple linear PD controller is not able to affect the nonlinear behaviors of the dynamics and stabilize the overall tracking procedure. As shown in Fig. 1, we proposed to use an feedback linearization controller to account for the nonlinear dynamics of the rope. The basic idea of the controller is to first transform the nonlinear dynamics to a linear system and then apply linear control law to stabilize the system.

In the physics engine, the rope is modeled as a chain of linked capsules. Each capsule is connected by a 2 DOF spherical joint (without twist motion). Therefore, the rope dynamics have the same form as the standard multi-joint robot.

$$\tau = M(q,\dot{q})\ddot{q} + c(q,\dot{q})\dot{q} + g(q) + \tau_{ext} \tag{1}$$

(1) shows the rope dynamics in joint space, where q, τ, M, c, g, and τ_{ext} represent the joint angle, joint force, inertia matrix, coriolis term, gravity, and the external force respectively.

The joint angle of the rope can be obtained via a simple 'inverse kinematics' of the chain, and all other terms can be easily obtained directly from the simulator. Then the feedback linearization control law is given by,

$$\tau = M(q,\dot{q})\ddot{q}_d + c(q,\dot{q})\dot{q} + g(q) + K_p(q_d - q) + K_d(\dot{q}_d - \dot{q}) \tag{2}$$

where K_p, K_d are tuned PD parameters, q_d is the desired joint position.

Compared with previous works, the feedback linearization controller is able to cancel the nonlinearity of the rope dynamics by the first three terms in (2). Then by properly selecting K_p and K_d, we are able to let the simulated rope track the desired point cloud and stabilize the entire procedure.

IV. Preliminary Experiments

To test the proposed rope tracking framework, several representative rope tracking tasks were conducted. Preliminary comparisons between our proposed framework with previous works are shown in this chapter. The experimental videos can be found at [1].

Shown in Figs. 2, 5, and 6, a 1-meter-long red rope was placed on a green table. A Microsoft Kinect (version 2) was utilized to get point cloud of the rope. The simulated rope in the Bullet Physics Engine was modeled by fifty linked capsules with density $1.5g/cm3$. The stiffness gain K_p and damping gain K_d were set as $10N/m$ and $0.5Ns/m$ respectively in the feedback linearization controller.

During the experiment, we manipulated the rope at a moderate speed. Fig. 5 shows the tracking results using the previous work [11], which do not have point cloud recovery module and feedback linearization controller. The tracking result is not stable due to missing point cloud. Fig. 6 shows the tracking result under occlusion with our proposed framework. A large portion of point cloud is missing due to occlusions. With the point cloud recovery module, we can recover most of the point cloud as the middle row images show, which improves the performance of node registration. Some point cloud, however, may still be missing when the rope is moving under occlusion even with the point recovery module. Such is a case when human holds the rope in hands as Fig. 6(c) shows. In this case, the foreground mask does not have historical information to recover the points until the occluded part is exposed to the Kinect again.

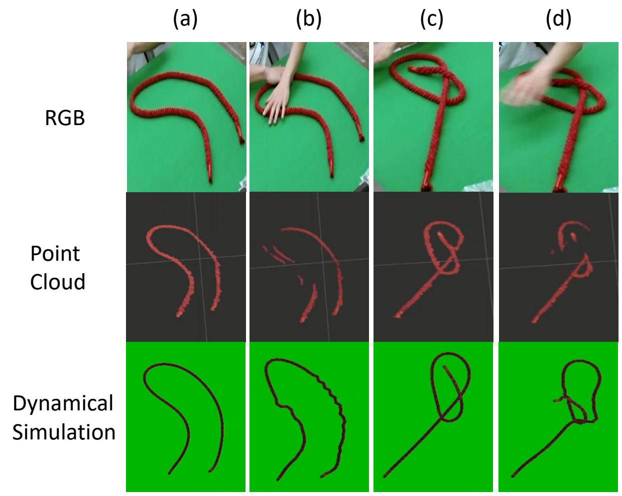

Fig. 5. Without point cloud recovery. Waving hands above the rope without touching the rope. Registration fails and the simulated rope deforms to unexpected shapes due to point cloud missing.

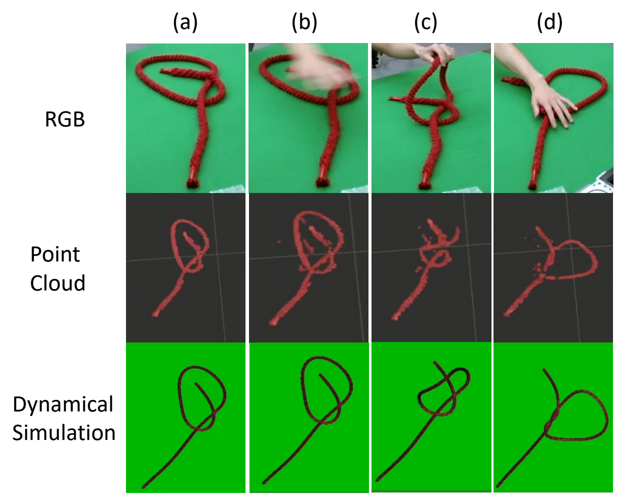

Fig. 6. With point cloud recovery. Tracking is robust to large occlusion.

V. Future Works and Discussion

This paper proposed a robust framework for tracking deformable objects, which includes point cloud recovery, node registration, feedback linearization controller, and dynamical simulation modules. Compared with previous works, the point cloud recovery step is able to robustly provide a complete point cloud of the object even under massive occlusions. In addition, the feedback linearization controller is able to stabilize the rope tracking procedure via first canceling higher-order terms in the dynamic equation and then using an additional PD control law to control the remaining linear dynamics. For future work, we would like to continue refining the overall structure of deformable objects manipulation to make the whole process more robust. In addition, more experiments on other challenging objects, such as clothes and sponges will be conducted.

REFERENCES

[1] Experimental Videos for Collision Avoidance Trajectory Optimization, https://changhaowang.github.io/DeformableTracking/RopeTracking.html.

[2] 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.

[3] T. Tang, C. Wang, and M. Tomizuka, "A framework for manipulating deformable linear objects by coherent point drift," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3426–3433, 2018.

[4] J. Schulman, A. Lee, J. Ho, and P. Abbeel, "Tracking deformable objects with point clouds," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1130–1137.

[5] S. Jin, C. Wang, and M. Tomizuka, "Robust deformation model approximation for robotic cable manipulation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6586–6593.

[6] 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," 2017.

[7] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," 2020.

[8] 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," 2020.

[9] A. Wang, T. Kurutach, K. Liu, P. Abbeel, and A. Tamar, "Learning robotic manipulation through visual planning and acting," 2019.

[10] C. Chi and D. Berenson, "Occlusion-robust deformable object tracking without physics simulation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6443–6450.

[11] T. Tang and M. Tomizuka, "Track deformable objects from point clouds with structure preserved registration," *The International Journal of Robotics Research*, 2018.

[12] T. Tang, C. Liu, W. Chen, and M. Tomizuka, "Robotic manipulation of deformable objects by tangent space mapping and non-rigid registration," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2689–2696.