Robotic Manipulation of Deformable Objects by Tangent Space Mapping and Non-Rigid Registration

Te Tang[1], Changliu Liu[1], Wenjie Chen[2] and Masayoshi Tomizuka[1]

***Abstract*— Recent works of non-rigid registration have shown promising applications on tasks of deformable manipulation. Those approaches use thin plate spline-robust point matching (TPS-RPM) algorithm to regress a transformation function, which could generate a corresponding manipulation trajectory given a new pose/shape of the object. However, this method regards the object as a bunch of discrete and independent points. Structural information, such as shape and length, is lost during the transformation. This limitation makes the object's final shape to differ from training to test, and can sometimes cause damage to the object because of excessive stretching. To deal with these problems, this paper introduces a tangent space mapping (TSM) algorithm, which maps the deformable object in the tangent space instead of the Cartesian space to maintain structural information. The new algorithm is shown to be robust to the changes in the object's pose/shape, and the object's final shape is similar to that of training. It is also guaranteed not to overstretch the object during manipulation. A series of rope manipulation tests are performed to validate the effectiveness of the proposed algorithm.**

I. INTRODUCTION

Manipulation of deformable objects, such as tying a rope and folding clothes, is still a challenging task for robots. The major difficulty lies in that the initial state (pose/shape) of the object differs from training scene to test scene. Therefore the taught manipulation trajectory at training cannot be directly applied during the test. A transformation procedure on the manipulation trajectory is required.

Schulman et al. [1] showed that existing non-rigid registration methods in the field of computer vision could be utilized on this robotics problem. Specifically, they used the thin plate spline-robust point matching (TPS-RPM) algorithm [2] to find a registration function f which mapped the training object and the test object, then transformed the training trajectory by the function f to get a new manipulation trajectory suitable for the test scenario. Several follow-up works [3]–[6] improved the original method by modifying the optimization objective or adding some more constraints in the TPS-RPM formulation. Most of them followed the same procedures: (1) find a registration function (2) warp the trajectory by the function. Both steps were performed in the Cartesian space.

However, this computer vision oriented method regards the physical object as a bunch of discrete and independent points, while dismissing the object's local physical properties, such as curvatures and distances between points. As a result, the transformed trajectory might not manipulate the object into a similar shape as shown at training. Overstretch and over-compression might also occur because the trajectory is generated without considering the object's physical limitations. In practice, these accidents could cause serious damage to the object or to the robot.

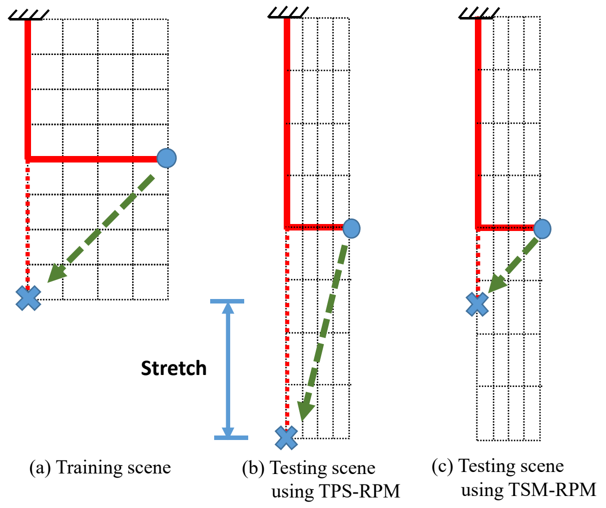

Stretch
(a) Training scene
(b) Testing scene using TPS-RPM
(c) Testing scene using TSM-RPM

Fig. 1. Rope manipulation by TPS-RPM and TSM-RPM. Red solid line shows the deformable rope, and green line is the manipulation trajectory. One end of the rope is fixed at the origin. (a) At training scene, the robot is taught to manipulate the L-shaped rope into a vertical straight line. (b) At test scene, TPS-RPM shrinks/extends the space in the horizontal/vertical direction, so as to register the training scene and the test scene. The training trajectory is warped in the same way to get the test trajectory, which overstretches the rope during manipulation. (c) The proposed TSM-RPM algorithm maps the scenes in the tangent space. The Cartesian space length is maintained during transformation. It manipulates the rope into a straight line without overstretching.

Take rope manipulation (Fig. 1) as an example. The initial shape of the rope is slightly changed from training scene to test scene. The TPS-RPM algorithm tries to register the two scenes by shrinking the Cartesian space in the horizontal directional while expanding in the vertical direction. As a result, it shrinks/expands the training trajectory in the same way to get the test trajectory (green line in Fig. 1(b)). This warped trajectory, however, violates the rope's length constraint and overstretches the rope during manipulation.

To deal with this problem, this paper proposes a new trajectory transformation method called tangent space mapping-robust point matching (TSM-RPM). Instead of mapping the training scene and the test scene in the Cartesian space, the registration is performed in the tangent space to maintain the

[1]Department of Mechanical Engineering, University of California, Berkeley, CA, USA. {tetang,changliuliu, tomizuka}@berkeley.edu

[2]Learning Robot Department, Robot Laboratory, FANUC Corporation, Japan. wjchen@berkeley.edu

structural information of the object. At test, the TSM-RPM algorithm has a better manipulation performance compared to TPS-RPM. It is able to manipulate the object into a similar shape as training, and is guaranteed not to overstretch the object during manipulation.

The remainder of this paper is organized as follows: Section II introduces the previous works of deformable manipulations, with a focus on the TPS-RPM methods. Section III describes the TPS-RPM algorithm for non-rigid registration, and the trajectory generation procedure by non-rigid transformation. In Section IV, the TSM-RPM algorithm is introduced in details. An invariance theorem is proposed and the object's length is proved to be constant during manipulation. Section V compares TPS-RPM and TSM-RPM by a series of rope manipulation tests, which validate the effectiveness of the proposed method. Supplementary videos can be found at [7]. Section VI concludes the paper and proposes future works.

II. Related Work

Manipulation of deformable objects has been a subject of robotics research for decades. Researchers have addressed this problem with many different methods, such as motion planning [8]–[10], knot theory [11]–[13] and multi-finger robotic hands [14], etc.

Moll et al. [8] constructed a motion planner for one-dimensional deformable objects by minimal energy optimization. They used the minimal energy model to estimate the dynamics of the rope and plan the trajectory with endpoint constraints. Morita et al. [11] developed a Knot Planning from Observation (KPO) system which estimated the states of the rope from the vision feedback by knot theory. The primitive movements were also predefined to change the states of the rope. Kudoh et al. [14] built a multi-finger hand and programmed skill motions by observing human knotting procedures. They realized the three dimensional in-air knotting with different types of knots.

Many of these methods, however, require empirical laws and are developed for a specific task, which is not easy to generalize for other tasks. For generalization, Schulman et al. [1] proposed a non-rigid registration method to teach the robot to manipulate the deformable objects by human demonstration. As described in Sectoin I, their method used TPS-RPM to transfer the original trajectory (taught by human demonstration) to get a new manipulation trajectory which was suitable for the test scene. This method has been successfully implemented on several tasks, such as tying ropes, folding clothes and opening bottles with various initial states. Since then, many follow-up works improved the non-rigid registration based method. Lee et al. [3] extended Schulman's approach by jointly optimizing the non-rigid registration and the trajectory optimization into a single optimization framework, such that the resulting trajectory is smoother given obstacles. Lee et al. [6] then incorporated the normals into the objective function to find a better registration to ensure that the robot gripper is vertical to the operation surface. Huang et al. [4] proposed an approach to learn the relevant appearance information of the deformable object by deep learning, and to use this additional information to improve the quality of non-rigid registration.

III. Deformable Manipulation by TPS-RPM

A. Non-rigid Registration

Non-rigid registration [15] is a class of methods that register two point clouds with non-rigid deformation. There are two problems to be addressed: given two point clouds, what is the correspondence between the points in one cloud to those in the other, and what is the proper transformation function such that after transformation, the distance between corresponding points is minimized.

The TPS-RPM algorithm is one of the well-known non-rigid registration methods which could approach to the local optimum of registration efficiently. Suppose there are two objects described by point clouds X and Y, consisting of $\{x_i \in \mathbb{R}^n, i = 1,2,\cdots,N\}$ and $\{y_j \in \mathbb{R}^n, j = 1,2,\cdots,N'\}$ in n dimensional space. In TPS-RPM, the non-rigid transformation function f is constructed in a specific form of thin-plate-spline (TPS [16]). The correspondence between X and Y is denoted by a correspondence matrix $M \in \mathbb{R}^{(N+1)\times(N'+1)}$. Each element $m_{ij} \in [0,1]$ indicates if point x_i corresponds to point y_j. $m_{ij} \to 0$ means low correspondence, while $m_{ij} \to 1$ represents high correspondence. The extra $(N+1)$th row and $(N'+1)$th column of M are introduced to handle outliers.

The objective of TPS-RPM is to find the optimal M and f that minimize the following fuzzy assignment-least squares energy function [17],

$$\begin{aligned}\min_{M,f} \quad & \sum_{i=1}^{N}\sum_{j=1}^{N'} m_{ij}\|y_j - f(x_i)\|_2^2 + \lambda\|f\|_{\text{TPS}}^2 \\ & + T\sum_{i=1}^{N}\sum_{j=1}^{N'} m_{ij}\log m_{ij} \qquad (1)\end{aligned}$$

$$\text{s.t.} \quad \sum_{i=1}^{N+1} m_{ij} = 1, \sum_{j=1}^{N'+1} m_{ij} = 1, m_{ij} \geq 0 \qquad (2)$$

where $\|f\|_{\text{TPS}}^2$ measures the curvature of the thin-plate-spline,

$$\|f\|_{\text{TPS}}^2 = \int dx \|D^2 f(x)\|_{\text{Frob}}^2 \qquad (3)$$

In the objective function (1), λ controls the trade-off between the residual of the assignment and the smoothness of the function. T is called the temperature parameter and controls the fuzziness of the correspondence. In general, smaller T forces the correspondences to be more binary and less fuzzy.

The above formulation consists of two interlocking optimization problems: a fuzzy assignment problem on the correspondence matrix M and a least-square problem on the transformation function f. It is their combination that makes the non-rigid registration problem difficult. At the expense of only getting the local optimum, Chui et al. [2] solved this problem by an alternating optimization algorithm, which successively updated M and f separately while gradually reducing λ and T.

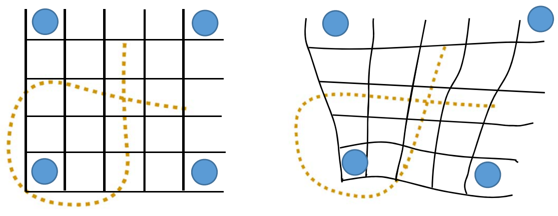


Fig. 2. Trajectory transformation by TPS-RPM on a 2-D example. Figure courtesy of [1]. The transformation function f is regressed such that the blue point clouds in the training scene and the test scene are mapped to each other in the Cartesian space. The training trajectory is then warped by the same function f to get the the manipulation trajectory used at test time.

The details of the TPS-RPM algorithm is as follows.

Step 1: Fix the transformation function f, update the correspondence matrix M by

$$m_{ij} = \frac{1}{T}\exp\left(-\frac{\|y_j - f(x_i)\|_2^2}{2T}\right) \tag{4}$$

and then iteratively normalize the row and column.

Step 2: Fix the correspondence matrix from Step 1, and update the transformation. Drop the terms independent of f in (1) to get the following least-square problem

$$\min_f \sum_{i=1}^{N}\sum_{j=1}^{N'} m_{ij}\|y_j - f(x_i)\|_2^2 + \lambda \int dx \|D^2 f(x)\|_{\text{Frob}}^2 \tag{5}$$

A simpler form is implemented instead for efficiency

$$\min_f \sum_{i=1}^{N} \|\tilde{y}_i - f(x_i)\|_2^2 + \lambda \int dx \|D^2 f(x)\|_{\text{Frob}}^2 \tag{6}$$

with $\tilde{y}_i = \sum_{j=1}^{N'} m_{ij} y_j$. Equation (6) has a closed form minimizer

$$f^*(x) = \sum_i a_i \sigma(x - x_i) + B^T x + c \tag{7}$$

where σ is the kernel function. Details of solving the minimizer's parameters can be found in [16].

After each iteration, parameters λ and T are gradually reduced according to a decreasing annealing schedule. The process is repeated until the lower bound of T is reached. The basic idea behind this heuristic is that more global and rigid transformations should first be favored before allowing more local non-rigid transformations.

B. *Trajectory Transformation in Cartesian Space*

By TPS-RPM, the point cloud of the deformable object at training and that at test could be registered by the transformation function f. The same function is then utilized to warp the demonstrated end-effector trajectory, so as to achieve a new trajectory suitable for the test scene (see Fig. 2). Suppose the demonstrated end-effector trajectory is represented as a series of poses $\{T_1, T_2, \cdots, T_T\}$, where each pose T_t consists

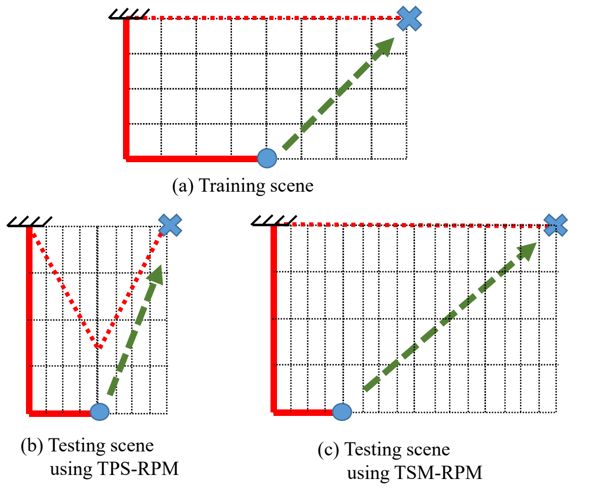

(a) Training scene
(b) Testing scene using TPS-RPM
(c) Testing scene using TSM-RPM

Fig. 3. Example of Rope manipulation by TPS-RPM and TSM-RPM. Red solid line shows the deformable rope, and green line is the manipulation trajectory. One end of the rope is fixed at the origin. (a) At training scene, the robot is taught to manipulate the L-shaped rope into a horizontal straight line. (b) At test scene, TPS-RPM shrinks/extends the space in the horizontal/vertical direction, so as to register the training scene and the test scene. The training trajectory is warped in the same way to get the test trajectory, which manipulates the rope into V-shape, not similar as training. (c) The proposed TSM-RPM algorithm maps the scenes in the tangent space. The curvature information is maintained during transformation. It manipulates the rope into a straight line similar as training.

of a position term p_t and an orientation term R_t. Then the trajectory can be transformed by

$$p_t^{\text{test}} = f(p_t^{\text{train}}) \tag{8}$$

$$R_t^{\text{test}} = \text{orth}\left(J_f(p_t^{\text{train}}) \cdot R_t\right) \tag{9}$$

where $J_f(p)$ is the Jacobian matrix of f evaluated at position p and $\text{orth}(\cdot)$ is a function that orthogonalizes matrices. Note that if f is a rigid transformation T_f, then this trajectory transformation procedure is equal to left-multiplying each end-effector pose T_t by T_f.

This trajectory transformation method by TPS-RPM has been successfully applied on many applications. However, the next section will show that TPS-RPM has limitation in keeping the object's structural information, and thus a new algorithm TSM-RPM is proposed.

IV. Manipulation by Tangent Space Mapping

A. *Tangent Space Mapping*

For a deformable manipulation task, the object is not only desired to move to a specific position or orientation, but also expected to change the shape to a specific configuration. Shape of the entire body is locally represented by the curvatures of the object points. Therefore the structural information which describes the relationship between one point and its surrounding points is important for the manipulation of the deformable object.

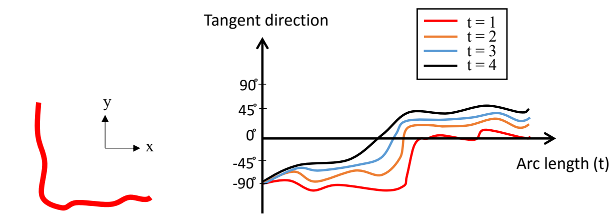
However, TPS-RPM regards the deformable object as a bunch of discrete and independent points. It only takes the points' position information into account, while dismissing the points' interrelationship, such as curvatures and local distances. This fact limits the TPS-RPM method from being able to manipulate the object under physical constraints (details in Fig. 1) as well as manipulating the object into a similar configuration as demonstrated at training (details in Fig. 3).

To address this problem, this section introduces a new trajectory transformation method called tangent space mapping-robust point matching (TSM-RPM).

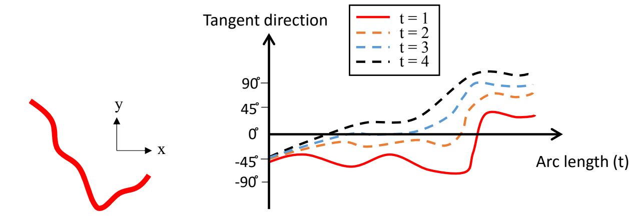
For the ease of explanation, here we take rope manipulation on a plane as an example. As shown in Fig. 4, the rope can be equivalently represented in the Cartesian space and in the tangent space. Fig. 4(a) and Fig. 4(c) show the rope at training scene and at test scene in the Cartesian space. Fig. 4(b) and Fig. 4(d) are the tangent graphs of the rope, where the horizontal axis is the arc length along the rope, and the vertical axis is the direction of the unit tangent vector. In this case, the rope's unit tangent vector is one dimensional.

During training, the rope is manipulated from the initial shape to the final shape. This training procedure can be decomposed into several snapshots at different time frames. At each time frame, a tangent graph of the rope can be constructed (see Fig. 4(b)). At test time, the rope starts with a different initial shape and consequently, a new initial tangent graph. A transformation function f_{TSM} could be found in the tangent space that maps the initial tangent graph at training to the initial tangent graph at test. That same function f_{TSM} can be utilized to warp the tangent graphs at training to get the corresponding tangent graphs at test in subsequent time frames.

After getting the tangent information of the rope at test, the tangents are integrated along the arc length to convert the tangent information into position information in order to get the manipulation trajectory that robot should follow at test time.



(a) Initial Shape of the Training Rope in Cartesian space (b) Tangent Graphs of the Training Rope



(c) Initial Shape of the Test Rope in Cartesian space (d) Tangent Graphs of the Test Rope

Fig. 4. Rope in the Cartesian space and in the tangent space. $t = 1,2,3,4$ represents four time frames of rope manipulation. $t = 1$ is the initial time, $t = 4$ is the final time.

B. TSM-RPM Algorithm

This subsection introduces the mathematical formulation of the TSM-RPM algorithm. Some notations of the tangent vectors are built up first. For a d-dimensional object living in n-dimensional space, $\phi_i^t \in \mathbb{R}^{n-d}$ denotes the training object's tangent vector at point i and at time frame t. Similarly, the test object's tangent vector can be represented by $\psi_j^t \in \mathbb{R}^{n-d}$, where j is the point index of the test object. The TSM-RPM algorithm is formulated as follows.

Step 1: Run TPS-RPM algorithm in the tangent space

(1) During training, extract the object's tangent vectors ϕ_i^t at each point i and each time frame t. At test, extract the test object's initial tangent vector $\psi_j^{t=1}$ at each point j.

(2) Calculate the tangent space correspondence matrix M and non-rigid transformation function f_{TSM} that register $\{\phi_i^{t=1}, i = 1,2,\cdots,N\}$ to $\{\psi_j^{t=1}, j = 1,2,\cdots,N'\}$ by iteratively solving Equations (4) and (7).

Step 2: Run tangent space mapping (TSM)

(1) Calculate the tangent vectors at test scene by non-rigid transformation from the second time frame $t = 2$ to the last time frame $t = t^{final}$

$$\psi_j^t = \sum_{i=1}^{N} M_{ij} \cdot f_{TSM}(\phi_i^t) \qquad \forall t = 2,\cdots,t^{final} \tag{10}$$

(2) Calculate the grasping point at test. Suppose the robot grasps the point i_G of the object at training, then the grasping point at training can be calculated by the correspondence matrix

$$j_G = \arg\max_j M_{(i_G,j)} \tag{11}$$

(3) Integrate the test tangents to get the Cartesian space position of grasping point at each time frame

$$p_{j_G}^t = \sum_{j=1}^{j_G} \Psi_j^t \cdot \delta_j \tag{12}$$

where δ_j is the distance between neighbouring points. Note that $\Psi \in \mathbb{R}^n$ is an unit vector, equivalent to expressing ψ in n-dimensional space. Ψ is utilized here so that the dimension is consistent to the Cartesian space points. The new manipulation trajectory is given by $\{p_{j_G}^t, t = 1,2,\cdots,t^{final}\}$

In practice, it is found that the correspondence matrix M' that register the two point clouds in the Cartesian space can directly serve as a good tangent space transformation. This makes sense because the corresponding points of two objects in the Cartesian space usually share the similar tangent features. This special transformation is formulated as

$$\psi_j^t = \sum_{i=1}^{N} M'_{ij} \cdot \phi_i^t \qquad \forall t = 2,\cdots,t^{final} \tag{13}$$

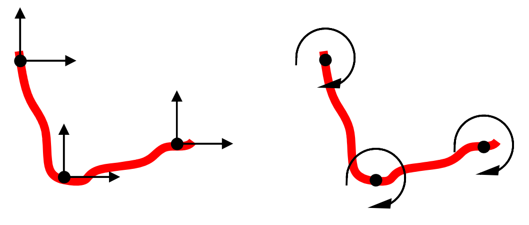


Fig. 5. Illustration of the difference between TPS-RPM and TSM-RPM. TPS-RPM tries to reallocate the positions of each points on the rope, which can easily lead to overstretching and thus cause damage. TSM-RPM tries to reassign the joint angles of the rope. No matter how much each joint is twisted, the rope's original length is maintained.

C. *Invariance Theorem of the TSM-RPM algorithm*

Objects like ropes can be considered as curves in differential geometry. This subsection will show that transformations under TSM-RPM maintain the curve's structural information and thus keep the curve's length invariant during manipulation.

Theorem 1: The length of the curve is invariant under the manipulation generated by TSM-RPM.

Proof: Assume the initial length of the curve at test is $L_{\text{test}}^{t=1}$. The length becomes L_{test}^{t} at time frame t. For any tangent vector $\Psi^t \in \mathbb{R}^n$ at time frame t, we have

$$|\Psi^t| = 1 \tag{14}$$

Therefore,

$$\begin{aligned} L_{\text{test}}^{t} &= \int_0^{L_{\text{test}}^{t=1}} |\Psi^t| d\tau \\ &= \int_0^{L_{\text{test}}^{t=1}} 1 \cdot d\tau \\ &= L_{\text{test}}^{t=1} \end{aligned} \tag{15}$$

where $d\tau$ is the differential arc length of the curve. ■

Equation (15) indicates the length of the test curve always keeps the initial value $L_{\text{test}}^{t=1}$, which is independent of the training curve's length L_{train}^{t}. This is a desired property since even if the curve length changes from training to test (for example, a long rope at training and a short rope at test), the algorithm will manipulate the test curve under the test length limitation despite of the training curve's length.

This invariance theorem can also be intuitively understood through Fig. 5. TPS-RPM regards the rope as a bunch of discrete points and tries to relocate the position of each point directly in the Cartesian space, which can lead to overstretching the rope during manipulation. In contrast, the TSM-RPM algorithm regards the rope as a long chain and instead rotates the joint angle of each chain in the tangent space. No matter how the angles are twisted, the rope's length does not change, i.e., length invariance. This is an important feature because otherwise safety cannot be guaranteed during robot manipulation.

To conclude, a new trajectory transformation method TSM-RPM is introduced. It is inspired by TPS-RPM, but different in the nature. The key difference is that the transformation takes place in the tangent space instead of Cartesian space in order to maintain the object's structural information.

V. Simulations and Results

To test the performance of the proposed TSM-RPM algorithm, several rope manipulation tests are performed and the results are analyzed in this section. Supplementary videos can be found at [7].

The manipulation tasks are simulated in V-REP [18] and the Bullet Physical Library [19] is selected as the physics engine. In these tests, two robot arms (FANUC LR Mate 200*i*D/7L) collaborate each other to manipulate a single red rope of length 40cm. The flexible rope is modelled as twenty 2cm long cylinders connected sequentially by spherical joints. At training, one robot arm fixes one end of the rope, while the other arm is taught by human to move the free end to manipulate the rope into a desired shape. At test time, the initial pose and shape of the rope are altered on purpose. The TSM-RPM and the TPS-RPM algorithms generate new manipulation trajectories given the test rope's initial states. The correspondence matrix M and transformation function f are calculated by solving (4) and (7) iteratively, with $T = 0.05$ and $\lambda = 1$. T and λ are decreased by multiplying an annealing rate of 0.93 after each iteration. Iteration terminates if $T \leq 10^{-5}$.

As shown in Fig. 6(a), the first test is to manipulate a curved rope into a straight line. The initial shape of the rope at training is like a 'L', of which the two sides have equal lengths. The test rope is also in L-shape, but one side is longer than the other. During the test scene, the manipulation trajectory (Fig. 6(b)) calculated by TPS-RPM moves the rope to the correct direction, but the final shape is curved and not similar to the one in training. The trajectory calculated by TSM-RPM (Fig. 6(c)) not only moves the rope to the right direction, but also manipulates to accurately reflect what was demonstrated in training.

The second test is to wind the red rope around a blue shaft. It is desired to wind the rope tightly around the shaft without overstretching the rope during manipulation. Fig. 7 shows a segment of the winding task. The robot is taught to bend the rope ninety degrees at training phase (Fig. 7(a)). At test scene, the initial shape of the rope is changed. TPS-RPM generates a trajectory to bend the new rope (Fig. 7(b)). However, at time frame $t = 3$, overstretch occurs and the rope breaks at time frame $t = 4$. In contrast, the TSM-RPM bends the test rope tightly around the shaft without overstretching the rope (Fig. 7(c)).

These tests show that compared to TPS-RPM, TSM-RPM can manipulates the rope into a better final shape and does not violate the length constraint. To quantitatively compare TSM-RPM and TPS-RPM, a benchmark is designed as shown in Fig. 8, which tests the two algorithms' computation cost, the final shape quality, the stretch rate and the robustness to the initial pose/shape change. At training, the operator demonstrates a trajectory to manipulate the red rope

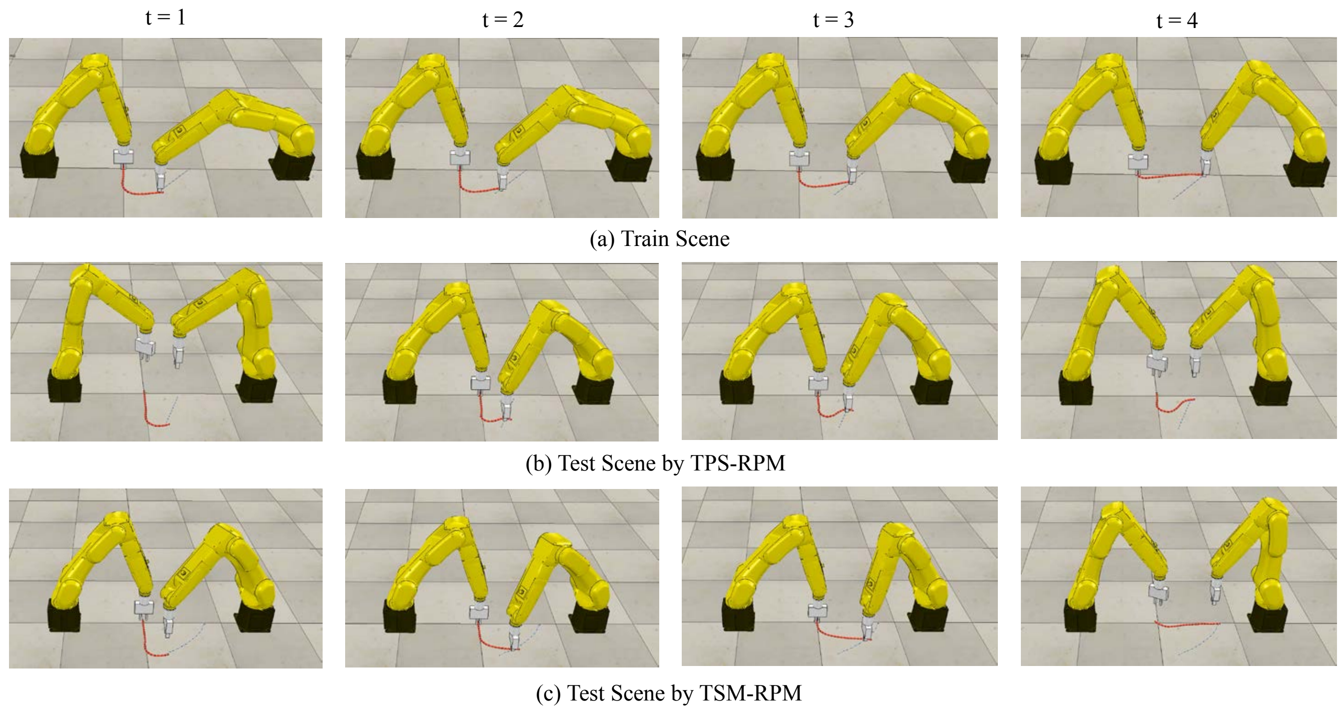


Fig. 6. Manipulating the rope into a straight line (horizontal direction) over four time frames, $t = 1, 2, 3, 4$. The initial shape of the rope is changed at test. TPS-RPM manipulates the test rope into a strange shape. TSM-RPM performs as well as training.

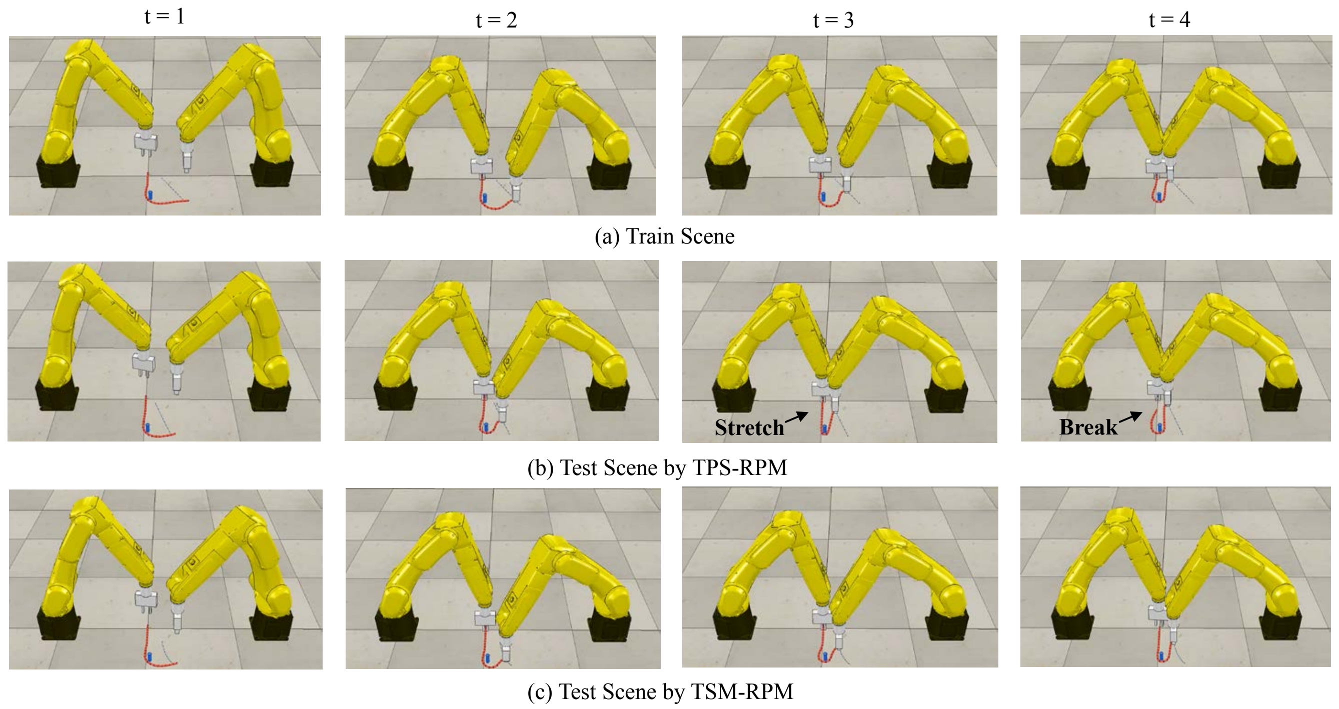


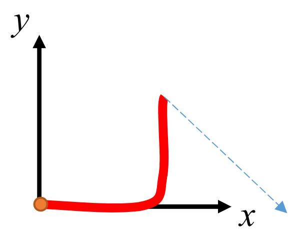
Fig. 7. Bending the rope ninety degrees around the shaft. The initial shape of the rope is changed at test. TPS-RPM overstretches the test rope at time frame $t = 3$ and the rope severs at $t = 4$. In contrast, TSM-RPM succeeds and does as well as training without incurring overstretching.

into a desired shape. At test, thousands of ropes with different poses/shapes are constructed by the following rules:

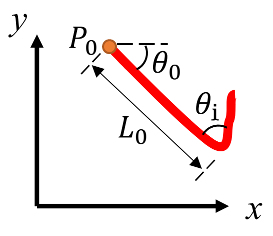
(1) the root point's position P_0 (in both the x and y directions) is changed from -20cm to +20cm, at increments of 10cm.

(2) The root point's orientation θ_0 is changed from $-45°$ to $+45°$, at increments of $15°$.

(3) The distance from the root point to the corner, L_0, is



(a) Rope at training



(b) Rope at test

Fig. 8. Test benchmark.

TABLE I
RESULTS OF ROPE MANIPULATIONS TEST BENCHMARK.

	Mean and Standard Deviation Registration Time(s)	Mean and Standard Deviation Shape Distortion(10^{-5})	# of Stretch
TPS-RPM	0.2735 ± 0.0093	2.915 ± 1.955	3968
TSM-RPM	0.2763 ± 0.0094	0.081 ± 0.261	0

changed from 10cm to 30cm, at increments of 4cm.

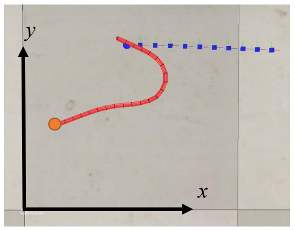
(4) The corner angle θ_c is changed from $+45°$ to $+135°$, at increments of $15°$

According to the construction rules, 7350 different ropes are constructed and manipulated by TPS-RPM and TSM-RPM respectively. Fig. 9 shows one trial of rope manipulations in the test benchmark.

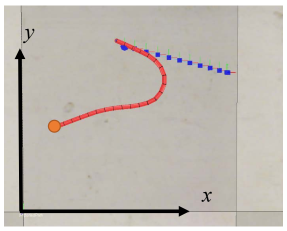
To quantify the similarity between the rope's final shape at training and at test, the non-rigid registration is performed again on the two final shapes, and the objective value in (1) is utilized to evaluate the shape distortion. The larger the distortion value, the more dissimilar between the two final shapes. Overstretch is detected by calculating the distance between the two grippers of the two robot arms. If the distance is larger than the rope length 40cm, the rope is overstretched.

Table I shows the results of the test benchmark. For each trial, the average registration time in TPS-RPM is 0.2735s. TSM-RPM takes 0.2763s. The increased 1.02% computation cost mainly comes from the reconstruction from the tangent space to the Cartesian space. For shape distortion, TPS-RPM has an average value of 2.915×10^{-5}, while TSM-RPM is 8.1×10^{-7}, which means the TSM-RPM has a better ability to manipulate the object into the configuration demonstrated in training. In 7350 trials, overstretch occurs 3968 times under TPS-RPM, while 0 time under TSM-RPM.

To conclude, these tests show that mapping in the tangent space make TSM-RPM have the ability to 'memorize' the shape of the object during training. At test, it will try to generate a trajectory which recovers the object to the configuration demonstrated at training. Therefore the final shape is usually as good as training. Moreover, as proven in invariance theorem, the manipulation trajectory generated by TSM-RPM will not exceed the length limitation of the object and therefore guarantee safety during manipulation.



(a) Trajectory generated by TPS-RPM



(b) Trajectory generated by TSM-RPM

Fig. 9. One trial of rope manipulations in the test benchmark.

VI. CONCLUSIONS AND FUTURE WORK

A TSM-RPM algorithm is proposed to teach the robot to manipulate deformable objects given various initial states. Compared to the TPS-RPM algorithm, TSM-RPM transforms the manipulation trajectory in tangent space instead of Cartesian space. Since the tangent describes the relation between one point and its surrounding points, mapping in the tangent space is able to maintain the object's local geometric properties. As a result, TSM-RPM can manipulate the test object into the similar configuration demonstrated at training time without fear of overstretching medium, as proven by the length invariance theorem. Therefore, overstretching does not occur and safety can be guaranteed during manipulation. A series of rope manipulation tasks have been tested in simulation to validate the performance of the proposed method. In the future, we would like to test the algorithm on more complicated experiments, such as rope knotting and unknotting. Also we will test on high dimensional objects such as clothes and soft sponges besides the rope.

ACKNOWLEDGMENT

We would like to thank FANUC Corporation, Japan for supporting this work. We also thank Prof. Pieter Abbeel and Sandy Huang for their insightful discussions and comments.

REFERENCES

[1] J. Schulman, J. Ho, C. Lee, and P. Abbeel, "Learning from demonstrations through the use of non-rigid registration," in *Proceedings of the 16th International Symposium on Robotics Research (ISRR)*, 2013.

[2] H. Chui and A. Rangarajan, "A new point matching algorithm for non-rigid registration," *Computer Vision and Image Understanding*, vol. 89, no. 2, pp. 114–141, 2003.

[3] A. X. Lee, S. H. Huang, D. Hadfield-Menell, E. Tzeng, and P. Abbeel, "Unifying scene registration and trajectory optimization for learning from demonstrations with application to manipulation of deformable objects," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 4402–4407.

[4] S. H. Huang, J. Pan, G. Mulcaire, and P. Abbeel, "Leveraging appearance priors in non-rigid registration, with application to manipulation of deformable objects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[5] A. X. Lee, A. Gupta, H. Lu, S. Levine, and P. Abbeel, "Learning from multiple demonstrations using trajectory-aware non-rigid registration with applications to deformable object manipulation," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 5265–5272.

[6] A. Lee, M. Goldstein, S. Barratt, and P. Abbeel, "A non-rigid point and normal registration algorithm with applications to learning from demonstrations," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, May 2015, pp. 935–942.

[7] Supplementary Video, http://me.berkeley.edu/~tetang/IROS2016/TSMRPM.html.

[8] M. Moll and L. E. Kavraki, "Path planning for deformable linear objects," *Robotics, IEEE Transactions on*, vol. 22, no. 4, pp. 625–636, 2006.

[9] M. Saha and P. Isto, "Motion planning for robotic manipulation of deformable linear objects," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006, pp. 2478–2484.

[10] T. Fukuda, T. Matsuno, and F. Arai, "Flexible object manipulation by dual manipulator system," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 1955–1960.

[11] T. Morita, J. Takamatsu, K. Ogawara, H. Kimura, and K. Ikeuchi, "Knot planning from observation," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 3. IEEE, 2003, pp. 3887–3892.

[12] T. Matsuno, D. Tamaki, F. Arai, and T. Fukuda, "Manipulation of deformable linear objects using knot invariants to classify the object condition based on image sensor information," *Mechatronics, IEEE/ASME Transactions on*, vol. 11, no. 4, pp. 401–408, 2006.

[13] J. Brown, J.-C. Latombe, and K. Montgomery, "Real-time knot-tying simulation," *The Visual Computer*, vol. 20, no. 2-3, pp. 165–179, 2004.

[14] S. Kudoh, T. Gomi, R. Katano, T. Tomizawa, and T. Suehiro, "In-air knotting of rope by a dual-arm multi-finger robot," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 6202–6207.

[15] W. R. Crum, T. Hartkens, and D. Hill, "Non-rigid image registration: theory and practice," *The British Journal of Radiology*, 2014.

[16] G. Wahba, *Spline models for observational data*. Siam, 1990, vol. 59.

[17] S. Gold, A. Rangarajan, C.-P. Lu, S. Pappu, and E. Mjolsness, "New algorithms for 2d and 3d point matching: Pose estimation and correspondence," *Pattern recognition*, vol. 31, no. 8, pp. 1019–1031, 1998.

[18] E. Rohmer, S. P. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 1321–1326.

[19] Bullet Tutorial, http://bulletphysics.org/wordpress.