2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Trajectory Planning under Different Initial Conditions for Surgical Task Automation by Learning from Demonstration

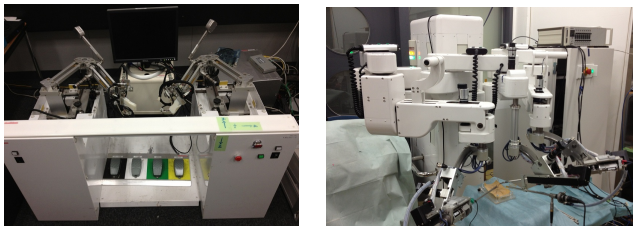
Takayuki Osa[1], Kanako Harada[1], Naohiko Sugita[1] and Mamoru Mitsuishi[1]

***Abstract*— The automation of surgical tasks has great potential for improving the performance of robotic surgery. The learning-from-demonstration approach has thus far been employed by many researchers when planning the trajectories of robotic instruments for automated surgical tasks. However, previous methods are applicable only when the demonstrations and trajectory generation are performed under the same initial conditions. In this paper, we propose an algorithm that learns through demonstrations and generates a trajectory regardless of the initial conditions. The variance of the demonstrated trajectories over the initial conditions is modeled and learned using a statistical method, and the learned trajectories are generalized and used to generate a trajectory. As an example, the bi-manual looping task of a surgical thread was demonstrated by changing the initial positions of the robot arms, and a trajectory was then planned given these initial arbitrary positions. The proposed algorithm was verified through simulations and experiments using an actual robotic surgical system.**

I. INTRODUCTION

In the last few decades, laparoscopic surgery has become a standard surgical procedure. Compared with conventional surgery, laparoscopic surgery reduces tissue trauma and recovery time because it is performed through small incisions made in the patient's body. On the other hand, laparoscopy imposes motion constraints on the surgical instruments used by the surgeon. As a result, laparoscopy can cause physical and mental fatigue to the surgeon.

In this context, robotic surgery has attracted great interest as a solution for the problems faced in laparoscopy. The most standard type of robotic surgical system is a master-slave system (e.g., the robotic system used in telesurgery, shown in Fig. 1), where an operator inputs motions at the master site, and a slave manipulator executes the input motion. Many studies have revealed that robotic surgery can provide dexterity and an intuitive interface to the surgeon. One of the most successful robotic surgery systems, i.e., the *da Vinci* system (Intuitive Surgical Inc., CA, USA), has been introduced in numerous hospitals worldwide [1]. Although robotic surgery has already witnessed significant advancements, potential improvements in the quality of surgical operations remain. One of the possible advancements in robotic surgery is the automation of surgical tasks. Such automation can reduce the physical and mental fatigue of the surgeon and reduce the surgery time.



(a) (b)

Fig. 1. Robotic telesurgery system: (a) master and (b) slave sites

Fig. 2. Examples of surgical instrument motions for a knot-tying task. Trajectories have to be adapted to the relative positions of each instrument.

In the last decade, many studies have focused on the automation of surgical tasks [2], [3], [4], [5]. Most of these studies are focused on the trajectory planning of surgical tasks requiring complex motions. As a master-slave surgical robotic system is capable of recording the trajectories, the learning-from-demonstration approach has been used to study the automation of surgical tasks. However, most of these previous studies cannot be applied to planning trajectories under different initial conditions, which is an essential function for practical use. For example, surgical knot-tying requires the collaborative motion of the left and right robot arms for making a double loop around one arm with a thread held by the other arm. This type of task requires planning the trajectories according to the initial relative positions of the left and right arms (see Fig. 2). However, this kind of trajectory planning is not trivial since the topological features of the learned trajectories need to be maintained after adaptation to different conditions. Although some studies have focused on generalizing learned trajectories for different conditions [2], [3], [6], they have certain limitations in terms of the computational costs or the modeling of the time-dependency of the trajectories.

In this paper, we propose a scheme to learn the trajectories of tasks from demonstrations and generalize a trajectory for different initial conditions. In our scheme, the trajectories are demonstrated and recorded under various initial condi-

This work was supported in part by the Global COE Program, Global Center of Excellence for Mechanical Systems Innovation by the Ministry of Education, Culture, Sports, Science and Technology, Grant-in-Aid for Scientific Research(S) 23226006 and Grant-in-Aid for JSPS fellows Number 25-7106.

[1]T. Osa, K. Harada, N. Sugita and M. Mitsuishi are with Department of Mechanical Engineering, the University of Tokyo, 7-3-1 Hongo Bunkyoku Tokyo, 113-8656 Japan, (Phone: +81-3-5841-6357; fax:+81-3-5841-6357; e-mail: osa— kanako — sugi — mamoru@nml.t.u-tokyo.ac.jp).

tions. By normalizing the trajectories along the time axis and modeling the variances of the trajectories over initial spatial conditions, we model both the time-dependency and the space-dependency of the trajectories. Furthermore, the computational cost of the proposed scheme is low enough to be applied to online trajectory planning.

Section II describes the differences between previously proposed methods and the method proposed in this paper. Section III describes the details of the proposed method. Section IV describes the experimental and simulation results. Section V presents a discussion of these results. Finally, section VI provides some concluding remarks regarding our proposal.

II. RELATED WORKS

Numerous studies have focused on learning from demonstration [7]. Several studies have also focused on the automation of surgical tasks. Berg et al. developed an automatic knot-tying system using the learning from demonstration approach, which allowed high-speed execution of the learned task [4]. However, their method is not applicable to generalization of trajectories to new initial conditions. Mayer et al. developed a scheme for adapting demonstrated trajectories to new situations using fluid dynamics [2], [3]. However, this method cannot be applied to online trajectory planning because of computational cost and limitations of modeling the situation. Schulman et al. proposed a related method [5], [8] that allows the generalization of demonstrated trajectories to new situations by computing a mapping from a demonstration situation onto a new situation. However, it is expected that the computational cost of this method will be too high for online trajectory planning.

In the field of humanoid automation, Gribovskaya et al. proposed a scheme for modeling trajectories as a nonlinear multivariate dynamical system [6]. This method can generalize trajectories to unseen conditions, but it cannot learn tasks if a robotic arm passes through the same area in a different direction, for example.

In our approach, we model the time- and space-dependency of demonstrated trajectories while a task is learned. The trajectories are normalized on the time axis using dynamic time warping (DTW), and the variance of the trajectories is modeled at each time step using locally weighted regression (LWR). Modeling using LWR allows important task features to be extracted from demonstrations. Moreover, the computational cost is sufficiently low for applying the model to online trajectory planning. The details of the proposed method are described in the next section.

III. METHOD

A. Overview of Proposed Method

An overview of the proposed method is outlined in Algorithm 1. First, the demonstrated trajectories are aligned along the time axis to deal with the variance of the demonstrated trajectories along the time axis. We assume that the jth demonstrated trajectories y_j are an *observation* of the reference trajectory z. This approach can be found in [9] and [4]. The reference trajectory is used as a norm of the demonstrated trajectories along the time axis. The time mapping of y_j to z is computed using DTW [10]. In the second step, the spatial variance of the time-normalized trajectories is modeled over the initial conditions using LWR. Here, the initial conditions refer to the initial state of the system, including the initial positions of the robotic surgical instruments and handled objects. The trajectory under a given new initial condition is estimated as a conditional expectation.

Algorithm 1 Overview of the proposed method

1. Initialize the time alignment:

$$\tau_i^j = i\frac{T^j}{N} \quad (i = 1, \ldots, N, j = 1, \ldots, M)$$

τ_i^j : time indices of the demonstrated trajectories of the jth trajectory at the ith step

2. Generate reference trajectories:

$$z \longleftarrow \textsc{KalmanSmoother}(y, \tau)$$

z : a reference trajectory, y : demonstrated trajectories

3. Normalize demonstrated trajectories in the time axis:

$$\tau_i^j \longleftarrow \textsc{DynamicTimeWarping}(z, y^j)$$

4. Model the spatial variance of trajectories with locally weighted regression (LWR)

$$\text{Compute } \mu_{\xi_{ini}},\ \mu_{\xi_s(\tau_i)},\ \sigma_{\xi_s(\tau_i)},\ \sigma_{\xi_s(\tau_i)\xi_{ini}} \text{ for } i = 1, \ldots, N$$

$\mu_{\xi_{ini}}$: mean of the initial conditions

$\mu_{\xi_s(\tau_i)}$: mean states of the time-normalized trajectories at the ith step

$\sigma_{\xi_s(\tau_i)}$: covariance matrix of the states of the time-normalized trajectories

$\sigma_{\xi_s(\tau_i)\xi_{ini}}$: covariance matrix of the initial conditions and states of the time-normalized trajectory

5. Estimate using the parameters from LWR

$$\hat{\xi}_s(\tau_i) \longleftarrow \arg\max P(\xi_s(\tau_i)|\xi_{ini,plan}) \text{ for } i = 1, \ldots, N$$

$\xi_{ini,plan}$: the given initial conditions $\hat{\xi}_s(\tau_i)$: trajectory planned for the initial condition, $\xi_{ini,plan}$

B. Normalization in the Time Domain

For normalization along the time axis, the method presented in [9] and [4] is used. In this method, the trajectories are modeled in discrete time. To represent the reference trajectory, we introduce the following notation:

$$z_i = \begin{bmatrix} \xi_i \\ \dot{\xi}_i \end{bmatrix} \tag{1}$$

where ξ_i is the state of the robotic system at the ith time step. The system is modeled as a linear system as follows:

$$z_{i+1} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} z_i + w_i,\ w_i \sim N\left(\mathbf{0}, \begin{bmatrix} P & 0 \\ 0 & Q \end{bmatrix}\right) \tag{2}$$

where P and Q are the noise terms.

First, the initial time mapping of the jth trajectory is set as follows:

$$\tau_i^j = i \cdot \frac{T^j}{N},\ (i = 0, \ldots, N) \tag{3}$$

where T^j is the time length of the jth trajectory, and N is the number of time steps in the reference trajectory. The demonstration y^j is regarded as an *observation* of the reference trajectory z. Therefore, the relationship between y^j and z can be expressed as follows:

$$\begin{bmatrix} y^1(\tau_i^1) \\ \vdots \\ y^M(\tau_i^M) \end{bmatrix} = \begin{bmatrix} I \\ \vdots \\ I \end{bmatrix} z_i + v_i,\ v_i \sim N\left(\mathbf{0}, \begin{bmatrix} R^1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & R^M \end{bmatrix}\right) \quad (4)$$

where R^j is the variance of the jth trajectory from the reference trajectory. Here, we set $R^1 = R^j = R^M = I$ because we equalize the contribution of each demonstration. Under this assumption, the Kalman smoother is employed to estimate the reference trajectory.

For normalizing the trajectories along the time axis, DTW is employed. DTW was originally developed for speech recognition [10], [11]. In DTW, a cost function that represents the distance between two sequences is computed [11]. The matching of the two sequences is obtained as a *warping path* that minimizes the total cost $c_p(X, Y)$, which is computed as follows:

$$c_p(X, Y) = \sum_{l=1}^{L} c(x_l, y_l) \quad (5)$$

where p is the *warping path*, L is the length of the warping path, and $c(x_l, y_l)$ is the local cost function. The total cost $c_p(X, Y)$ is minimized using a dynamic programming approach. Here, we computed the following equation recursively from $(n, m) = (N, N)$ to $(n, m) = (1, 1)$, where N is the length of the sequence.

$$D(n, m) = \min\{D(n-1, m-1), D(n-2, m-1), D(n-1, m-2)\} + c(x_n, y_m) \quad (6)$$

Although we have many other options for a recursive equation [10], the above equation worked well for matching the demonstrated trajectories to the reference trajectory in our system.

C. Modeling Variance of Trajectories over the Initial Conditions

LWR was employed to model the spatial variance of the time-normalized trajectories over the initial conditions [12]. Although Gaussian mixture regression (GMR) is a popular method for modeling a nonlinear distribution [13], [14], [6], its performance is affected by the number of samples and clusters used. In the learning-from-demonstration approach, the number of demonstrations can be limited; therefore, LWR, which works well with a relatively small number of samples, is used.

In the LWR process, we employed the following kernel function to compute the weight for each demonstration :

$$h_j = \begin{cases} 0 & (\left\| \xi_{ini}^j - \xi_{ini,plan} \right\|) > d \\ 1 & (\left\| \xi_{ini}^j - \xi_{ini,plan} \right\|) \leq d \end{cases} \quad (7)$$

where d is a constant value, ξ_{ini}^j is the initial condition under which the jth trajectory was demonstrated, and $\xi_{ini,plan}$ is the given initial condition for the trajectory planning. This kernel function excludes demonstrations from the learning if the initial conditions are very different from the given initial conditions. It also equally weighs the contributions from the demonstrations recorded under similar initial conditions.

Here, we denote the state of the robotic system at time t as $\xi_s(t)$. The mean of ξ_{ini}, the mean of $\xi_s(t)$, the variance of ξ_{ini}, and the covariance of ξ_{ini} and $\xi_s(t)$ can be computed as follows[12]:

$$\mu_{\xi_{ini}} = \frac{1}{n}\sum_{j=1}^{M} h_j \xi_{ini}^j, \quad \mu_{\xi_s(\tau_i))} = \frac{1}{n}\sum_{j=1}^{M} h_j \xi_s^j(\tau_i^j)) \quad (8)$$

$$\sigma_{\xi_{ini}} = \frac{1}{n}\sum_{j=1}^{M} h_j (\xi_{ini}^j - \mu_{\xi_{ini}})(\xi_{ini}^j - \mu_{\xi_{ini}})^T \quad (9)$$

$$\sigma_{\xi_{ini}\xi_s(\tau_i)} = \frac{1}{n}\sum_{j=1}^{M} h_j (\xi_{ini}^j - \mu_{\xi_{ini}})(\xi_s^j(\tau_i^j)) - \mu_{\xi_s(\tau_i))})^T \quad (10)$$

where $n = \sum_{j=1}^{M} h_j$ and τ_i^j is a normalized time-alignment computed using the method in the previous section. Thereafter, the trajectory for an initial state $\xi_{ini,plan}$ can be estimated as a conditional expectation as follows:

$$\hat{\xi}_s(\tau_i) = \mu_{\xi_s(\tau_i)} - \sigma_{\xi_{ini}\xi_s(i)}\sigma_{\xi_{ini}}^{-1}(\xi_{ini,plan} - \mu_{\xi ini}) \quad (11)$$

The whole trajectory for a given initial condition can therefore be obtained by computing Eq. (11) for $i = 0, \ldots, N$. After modeling the variance of the time-normalized trajectories by computing Eq. (10), the computation for planning a trajectory under a given initial state is relatively trivial because the necessary computation requires only Eq. (11). The proposed scheme can therefore be applied to online trajectory planning owing to its low computational cost.

IV. EXPERIMENT AND SIMULATION

The proposed trajectory planning scheme was verified experimentally and using simulations. The trajectories required for learning were demonstrated and recorded using a robotic surgery system (Fig. 1). Simulations were conducted to test the performance of the planned trajectories. The proposed scheme was then implemented in a robotic system, and the performance of the system was evaluated in experiments. The robotic system was developed for telesurgery, and its performance was verified through teleoperations [15]. In the robotic system, an operator input the requisite motions via a master manipulator, and a slave manipulator executed the motion signals sent from the master site.

In this experiment, the operator demonstrated three tasks - TOUCH AND RETURN, LOOPING WITH A SINGLE ARM, and LOOPING WITH TWO ARMS - and the trajectories of the slave manipulator were recorded. The motions of the right arm were learned in the TOUCH AND RETURN and LOOPING WITH SINGLE ARM tasks, and the motions of both the left and right arms were learned in the LOOPING WITH TWO ARMS task.

Collected data were processed and learned to create a model of each task. The demonstrations were performed

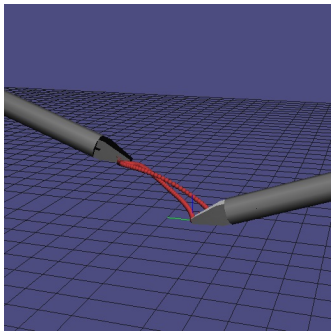
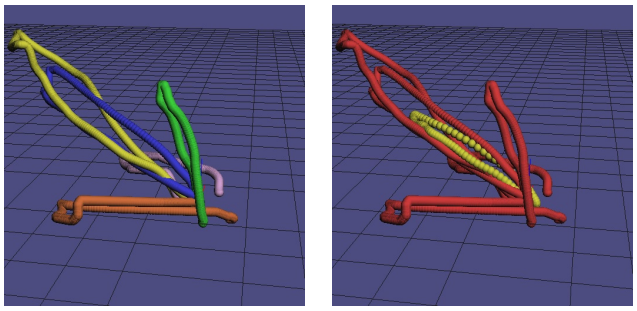
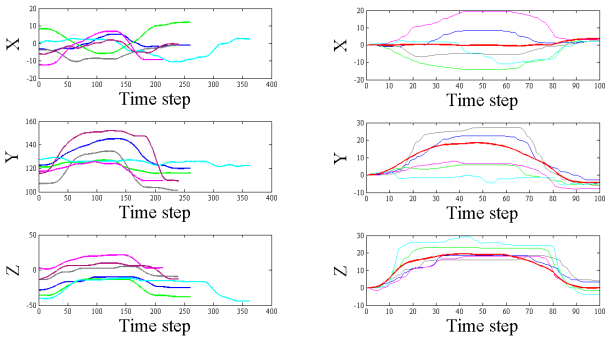


Fig. 3. Example of the trajectories demonstrated for the TOUCH AND RETURN task.



(a) (b)

Fig. 4. Dataset for learning the TOUCH AND RETURN task: (a) the colored dots represent trajectories demonstrated for the TOUCH AND RETURN task, and (b) the yellow dots represent the reference trajectory generated from the dataset shown in (a).



(a) (b)

Fig. 5. Results of DTW from the dataset in shown in Fig. 4: (a) the data before being processed using DTW, and (b) the data processed through DTW. The thick red lines show the norm of the demonstrated trajectory in the time domain, which is generated using the method described III-B.

nine times for each task. The demonstrations dataset was separated into eight demonstrations for learning the tasks and one demonstration for validating the planned trajectory, wherein the planned trajectories and the demonstrated trajectories were compared with the same initial conditions. To quantify the trajectory planning performance, the norm of the jerk, that is, $1/T\sum_{t=1}^{t=T}\left\|\dddot{\xi}_s\right\|$, for the planned trajectories and demonstrated trajectories under the same initial conditions and the root mean square (RMS) error, $1/T\sum_{t=1}^{t=T}\left\|\xi_s(t)-\hat{\xi}_s(t)\right\|$, between the planned trajectory and the demonstrated trajectory were computed.

In this experiment, the initial conditions referred to the position of the left arm relative to the right arm. Specifically, ξ_{ini} in (10) and (11) is replaced with the following equation:

$$\xi_{ini} = \xi_{s,l}(0) - \xi_{s,r}(0) \tag{12}$$

Here, $\xi_{s,l}(t)$ and $\xi_{s,r}(t)$ are the positions of the left and right arm, respectively, at time t. In the TOUCH AND RETURN and LOOPING WITH A SINGLE ARM tasks, the trajectory of the robotic system can be expressed as follows:

$$\xi_s(t) = \xi_{s,r}(t) \tag{13}$$

In the LOOPING WITH TWO ARMS task, the trajectory of the robotic system can be expressed as follows:

$$\xi_s(t) = [\xi_{s,r}(t), \xi_{s,l}(t)]^T \tag{14}$$

The left and right arms were positioned parallel to each other in opposite orientations to simplify the TOUCH AND RETURN and LOOPING WITH SINGLE ARM tasks. In the LOOPING WITH TWO ARMS task, the left and right arms were positioned such as observed frequently in robotic surgery.

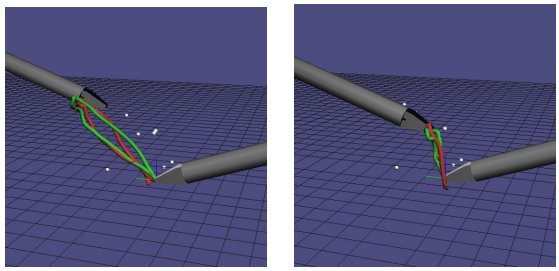
A. Trajectory Planning Simulation

1) TOUCH AND RETURN Task: The TOUCH AND RETURN task involved moving the right arm to touch the tip of the left arm and then moving it back to its initial position, as shown in Fig. 3. This task was designed to demonstrate trajectory generation for a time- and space-dependent task wherein the right arm needs to pass the same area in the opposite direction.

Figure 4 shows the demonstrated trajectories and the reference trajectory generated from the demonstration. The results of DTW are shown in Fig. 5. As shown in Fig. 5, the trajectories were normalized appropriately in the time domain, and, as a consequence, the phases of the motions were synchronized. The demonstration trajectories and trajectories planned under the same initial conditions are shown in Fig. 6. In this simulation, the computational time required for planning the trajectories was 60 - 80 ms using a 64-bit machine with an Intel Core i7-4600U CPU of 2.1 GHz. The averages of the norm of the jerk for the planned trajectories and demonstrated trajectories were 0.42 and 0.43, respectively [1]. The average RMS error between the planned trajectory and demonstrated trajectory was 6.4 [mm]. As shown in Fig. 6, the TOUCH AND RETURN task was generalized successfully under different initial conditions. In addition, the smoothness of the planned trajectory was comparable to that of the demonstrated trajectories.

2) LOOPING WITH A SINGLE ARM Task: The LOOPING WITH A SINGLE ARM task involved making a double loop around the left arm of the robot using a thread held in the right arm (Fig. 7). In the LOOPING WITH A SINGLE ARM task, the right arm was required to pass the same

[1]The unit of the norm of the jerk is omitted because the norm of the jerk was computed in the normalized time domain.



(a) (b)

Fig. 6. Examples of the trajectories generated for the TOUCH AND RETURN task under different initial conditions. The red and green dots represent the demonstrated trajectories for validation and the planned trajectories under the same initial conditions, respectively. The white dots represent the positions of the left arm in demonstrations used for learning the task.

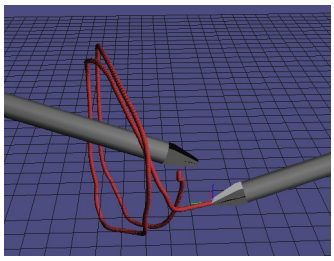
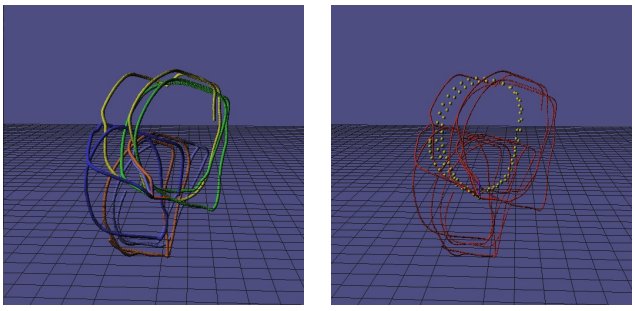


Fig. 7. Example of the trajectories demonstrated for the LOOPING WITH A SINGLE ARM task.

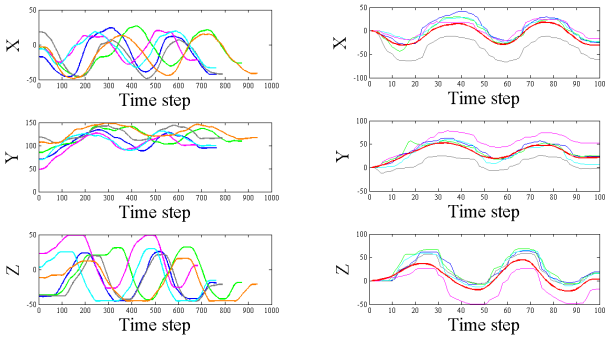
area twice in the same direction. In addition, the learned trajectories had to be adapted to avoid a collision between the two arms during the task.

The demonstrated trajectories and the reference trajectory generated from the demonstration are shown in Fig. 8. The results of DTW are shown in Fig. 9, and they indicate that the trajectories obtained from the demonstrations were normalized successfully in the time domain and that the phases of the motions of the normalized trajectories were synchronized with those of the reference trajectory. The demonstration trajectories and trajectories planned under the same initial conditions are shown in Fig. 10. The averages of the norm of the jerk for the planned trajectories and



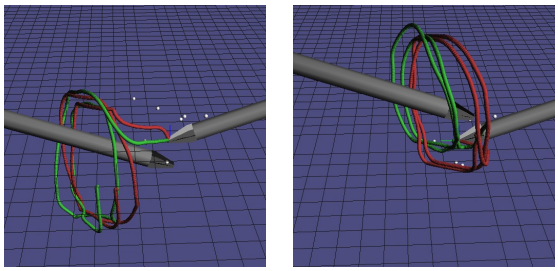
(a) (b)

Fig. 8. Dataset for learning the LOOPING WITH SINGLE ARM task: (a) the colored dots represent the trajectories used for learning the LOOPING WITH SINGLE ARM task, and (b) the yellow dots represent the reference trajectory generated from the dataset shown in (a).



(a) (b)

Fig. 9. Results of DTW from the dataset in shown in Fig. 8: (a) the data before being processed using DTW, and (b) the data processed through DTW. The thick red lines show the norm of the demonstrated trajectory in the time domain, which is generated using the method described in section III-B.



(a) (b)

Fig. 10. Examples of the trajectories generated for the LOOPING WITH A SINGLE ARM task under different initial conditions. The red and green dots represents demonstrated trajectories for validation and planned trajectories under the same initial conditions, respectively. The white dots represent the positions of the left arm in the demonstrations used for learning the task.

demonstrated trajectories were 1.49 and 1.51, respectively [2]. The average RMS error between the planned trajectory and demonstrated trajectory was 24.4 [mm]. In this simulation, the computational time required for planning the trajectories was 70 - 90 ms using a 64-bit machine with an Intel Core i7-4600U CPU of 2.1 GHz. The LOOPING WITH A SINGLE ARM task was generalized for different initial conditions, without collisions between the robot arms. Thanks to the statistic modeling, the planned trajectories do not have the same small deviations found in the demonstration trajectories.

3) LOOPING WITH TWO ARMS task: The LOOPING WITH TWO ARMS task involved making a double loop around one arm with a thread held by the other arm while moving both arms. During practical surgical knot-tying, both the right and left arms are moved to adjust the tension of the thread and to prevent it from slipping away from the arm. The LOOPING WITH TWO ARMS task was designed to demonstrate trajectory generation for the collaborative but autonomous motions of two arms, which requires more

[2]The unit of the norm of the jerk is omitted because the norm of the jerk was computed in the normalized time domain.

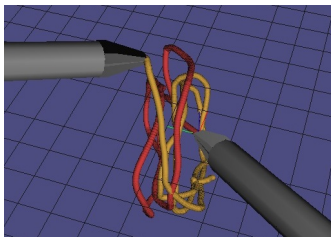
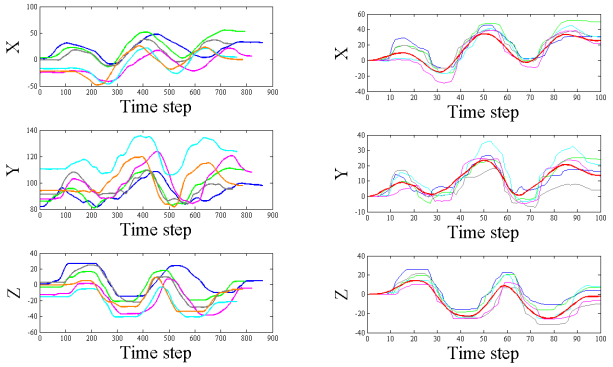


Fig. 11. Example of the trajectories demonstrated for the LOOPING WITH TWO ARMS task. The red and orange dots represent the trajectories of the left and right arms, respectively.



(a) (b)

Fig. 12. Results of DTW from the dataset for learning the LOOPING WITH TWO ARMS task: (a) the trajectories of the right arm before processed using DTW, and (b) the trajectories processed through DTW. The thick red lines show the norm of the demonstrated trajectory in the time domain, which is generated using the method described in section III-B.

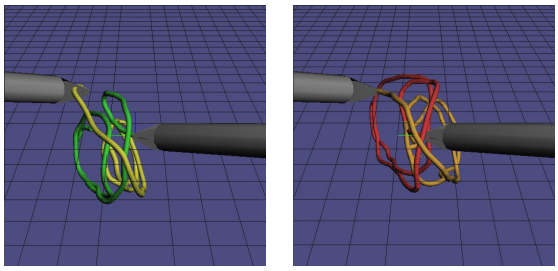
degrees of freedom than the other two tasks. Figure 11 shows an example of the trajectories demonstrated for the LOOPING WITH TWO ARMS task.

In the same manner as the other tasks, the demonstrated trajectories were normalized in the time domain (Fig. 12). Examples of the planned trajectory and the demonstrated trajectory for the LOOPING WITH TWO ARMS task under the same initial condition are shown in Fig. 13. In this simulation, the computational time required to plan the trajectories was 80 - 100 ms using a 64-bit machine with an Intel Core i7-4600U CPU of 2.1 GHz. The averages of the norm of the jerk for the planned trajectories and demonstrated trajectories were 1.76 and 1.89, respectively [3]. The average of RMS error between the planned trajectory and demonstrated trajectory was 24.3 [mm]. As shown in Fig. 13, the trajectories of the LOOPING WITH TWO ARMS task were planned successfully without any collisions, and the planned trajectories were smoother than the demonstrated trajectories.

B. Implementation in a Robotic System

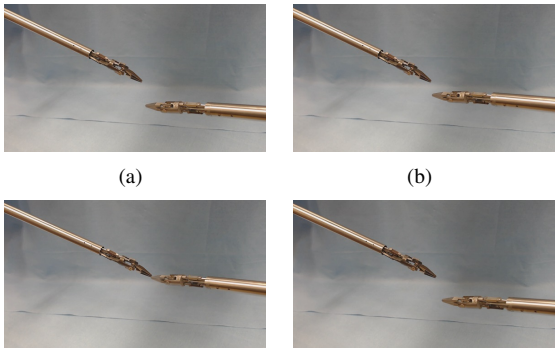
To implement the proposed method in the actual robotic system shown in Fig. 1, the planned trajectories had to be

[3]The unit of the norm of the jerk is omitted because the norm of the jerk was computed in the normalized time domain.



(a) (b)

Fig. 13. Trajectories for the LOOPING WITH TWO ARMS task. (a) the planned trajectory and and (b) the demonstrated trajectory for validation under the same initial condition. The white dots represent the positions of the left arm in demonstrations used for learning the task.



(a) (b)

(c) (d)

Fig. 14. Execution of TOUCH AND RETURN task. The task was executed from (a) to (d).

properly interpolated. In the developed system, the planned trajectories were interpolated using a cubic spline. The frequency of the control signal from the slave controller system to the motor drivers of the slave robot was fixed; thus, the density of the interpolation determined the execution speed of the planned trajectory, and the length of the planned trajectory determines the task completion time. In the implemented system, the planned trajectories were interpolated to allow automated tasks to proceed faster than manual tasks. More specifically, the interpolation density was determined, and therefore the autonomous task completion time does not surpass 80% of the shortest manual task completion time.

Figures 14, 15, and 16 show the TOUCH AND RETURN task, LOOPING WITH A SINGLE ARM task, and LOOPING WITH TWO ARMS task demonstrated using the actual robotic system, respectively. The system successfully performed each of the three tasks without any collisions. However, in the LOOPING WITH TWO ARMS task, the thread occasionally slipped away from the robotic surgical instrument when the thread was twisted. Therefore, the state of the thread should be considered when developing an autonomous knot-tying system for robotic surgery.

V. DISCUSSION

It was verified through simulations and experimentally that the proposed scheme can learn time- and space-dependent tasks. Although the initial condition, ξ_{ini}, in the simulations

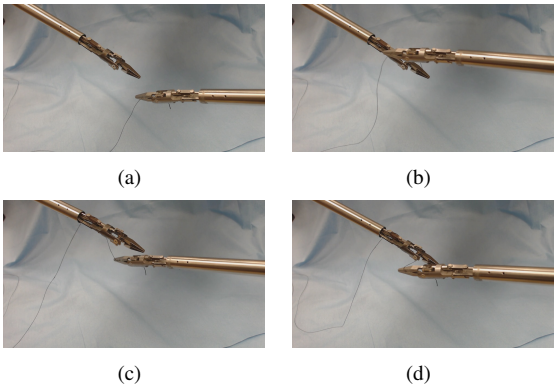


Fig. 15. Execution of the LOOPING WITH SINGLE ARM task. The task was executed from (a) to (d).

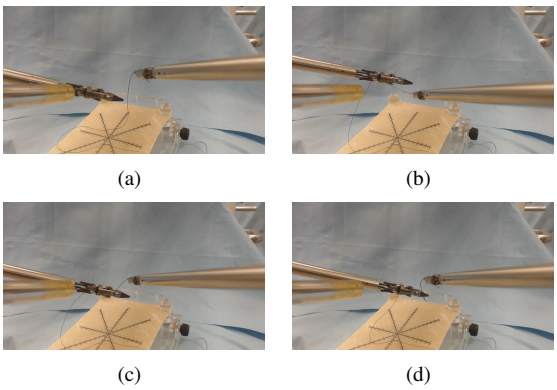


Fig. 16. Execution of LOOPING WITH TWO ARMS task. The task was executed from (a) to (d). Both the left and right arms moved automatically.

and experiments described in this study was the relative position of one arm to the other, the initial condition can be any set of parameters of the initial robotic state that can be obtained by the robotic system, such as the orientation angles, speed, and acceleration of the arms. The initial condition can include environmental parameters such as the position of the objects to be handled by the robot once the recognition of such an environmental state becomes feasible. Therefore, the proposed trajectory-planning scheme can be applied to many other applications. For example, when ξ_{ini} includes the position of a surgical needle, the proposed scheme can be applied to the automatic grasping of a surgical needle.

The proposed scheme models the variance of the demonstrated trajectories over the initial conditions linearly and locally, and therefore, it is valid when linear and local modeling applies. Hence, to achieve a satisfactory performance, the variance of the initial demonstration conditions should be sufficiently large.

VI. CONCLUSIONS

We developed a method for planning a trajectory under different initial conditions using the learning-from-demonstration approach. The developed method was verified through simulations and implementation in a telesurgical robotic system. The simulation and experimental results showed that the proposed algorithm successfully learned the given tasks and generalized them for different initial conditions. In the simulations and experiments, a time normalization of the given trajectories was performed through DTW. Trajectory planning under the given initial conditions was successfully performed by modeling the variance of the demonstrated trajectories using LWR. As a next step, the proposed scheme will be extended to handle environmental information and used to manipulate surgical tools such as surgical needles and threads.

REFERENCES

[1] G. Guthart and J. Salisbury, J.K., "The intuitivetm telesurgery system: overview and application," in *2000 IEEE International Conference on Robotics and Automation (ICRA) Proceedings*, vol. 1, 2000, pp. 618–621 vol.1.

[2] H. Mayer, I. Nagy, A. Knoll, E. Braun, R. Lange, and R. Bauernschmitt, "Adaptive control for human-robot skilltransfer: Trajectory planning based on fluid dynamics," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 1800–1807.

[3] H. Mayer, I. Nagy, D. Burschka, A. Knoll, E. Braun, R. Lange, and R. Bauernschmitt, "Automation of manual tasks for minimally invasive surgery," in *Autonomic and Autonomous Systems, 2008. ICAS 2008. Fourth International Conference on*, 2008, pp. 260–265.

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

[5] J. Schulman, A. Gupta, S. Venkatesan, M. Tayson-Frederick, and P. Abbeel, "A case study of trajectory transfer through non-rigid registration for a simplified suturing scenario," in *Proceedings of the 26th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[6] E. Gribovskaya, S. M. Khansari-Zadeh, and A. Billard, "Learning non-linear multivariate dynamics of motion in robotic manipulators," *I. J. Robotic Res.*, vol. 30, no. 1, pp. 80–117, 2011.

[7] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.

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

[9] P. Abbeel, A. Coates, and A. Y. Ng, "Autonomous helicopter aerobatics through apprenticeship learning," *Int. J. Rob. Res.*, vol. 29, no. 13, pp. 1608–1639, Nov. 2010.

[10] H. Sakoe and S. Chiba, "Dynamic programming algorithm for spoken word recognition," in *IEEE Transaction on Acoustics, Speech and Signal Processing*, A. Waibel and K.-F. Lee, Eds., San Francisco, CA, USA, 1978, pp. 159–165.

[11] M. Mueller, *Information Retrieval for Music and Motion*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[12] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," *Journal of Artificial Intelligence Research*, vol. 4, pp. 129–145, 1996.

[13] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 286–298, 2007.

[14] C. Reiley, E. Plaku, and G. Hager, "Motion generation of robotic surgical tasks: Learning from expert demonstrations," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, 2010, pp. 967–970.

[15] M. Mitsuishi, M. Hashizume, P. Navicharern, Y. Fujino, K. Onda, S. Yasunaka, N. Sugita, J. Arata, H. Fujimoto, K. Tanimoto, K. Tanoue, S. Ieiri, K. Konishi, and Y. Ueda, "A telesurgery experiment between japan and thailand," in *The 6th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2009)*, 2009.