Guiding Trajectory Optimization by Demonstrated Distributions

Takayuki Osa[1], Amir M. Ghalamzan E.[2], Rustam Stolkin[2], Rudolf Lioutikov[1], Jan Peters[1,3] and Gerhard Neumann[1]

***Abstract*—Trajectory optimization is an essential tool for motion planning under multiple constraints of robotic manipulators. Optimization-based methods can explicitly optimize a trajectory by leveraging prior knowledge of the system and have been used in various applications such as collision avoidance. However, these methods often require a hand-coded cost function in order to achieve the desired behavior. Specifying such cost function for a complex desired behavior, e.g., disentangling a rope, is a non-trivial task that is often even infeasible. Learning from demonstration (LfD) methods offer an alternative way to program robot motion. LfD methods are less dependent on analytical models and instead learn the behavior of experts implicitly from the demonstrated trajectories. However, the problem of adapting the demonstrations to new situations, e.g., avoiding newly introduced obstacles, has not been fully investigated in the literature. In this paper, we present a motion planning framework that combines the advantages of optimization-based and demonstration-based methods. We learn a distribution of trajectories demonstrated by human experts and use it to guide the trajectory optimization process. The resulting trajectory maintains the demonstrated behaviors, which are essential to performing the task successfully, while adapting the trajectory to avoid obstacles. In simulated experiments and with a real robotic system, we verify that our approach optimizes the trajectory to avoid obstacles and encodes the demonstrated behavior in the resulting trajectory.**

***Index Terms*—Motion and Path Planning, Collision Avoidance, Learning and Adaptive Systems, Manipulation Planning.**

I. INTRODUCTION

MOTION planning is an essential component of robotic systems. However, it is not trivial to plan robot motions to perform tasks successfully under various constraints, such as the kinematics of manipulators. In the field of motion planning, there are three promising categories: optimization-based approaches, sampling-based approaches, and learning from demonstration (LfD) approaches.

Sampling-based methods such as the Probabilistic Roadmap (PRM) and Rapidly exploring Random Tree (RRT) algorithms are used in various applications [1], [2]. Since sampling-based methods can deal with complex situations, they are often used for path planning of mobile robots. However, sampling-methods are computationally inefficient for motion planning of robotic manipulators [3], [4].

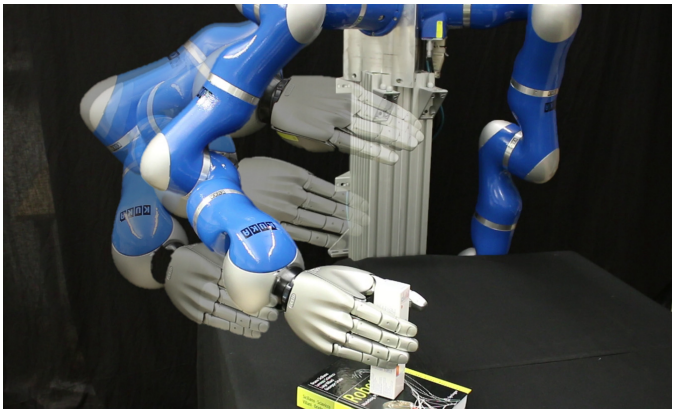


Fig. 1: Motion for grasping an object. Motion planning is an essential component of robotic systems.

Optimization-based methods such as Covariant Hamiltonian Optimization for Motion Planning (CHOMP) [3] and TrajOpt [4] are popular in the field of motion planning with robotic manipulators as they can efficiently deal with constraints, including obstacles and joint limits [3], [4]. These methods formulate motion planning as an optimization problem and can handle complex environments if the model of the environment is available. However, optimization-based methods require hand-coded cost functions to formulate the desired behavior, and it is often not practical to manually construct a task-specific cost function for complex robotic tasks, e.g., disentangling a rope.

Learning from demonstration (LfD), or programming by demonstration (PbD), offers an alternative, more intuitive way to specify the desired behavior as we can teach robotic motions simply by demonstrating the trajectories needed to execute the tasks [5], [6]. However, it is not feasible to demonstrate trajectories in all possible situations. Hence, additional processes are often necessary to adapt the predicted trajectory to the actual given situation. For example, if there are obstacles that were not present in the demonstrations, the trajectory prediction needs to be modified to avoid the obstacles. In order to address the motion planning with collision avoidance, it is desirable to develop a method that combines the advantages of optimization-based motion planning and LfD methods.

In this paper, we present a motion planning method based on functional gradient trajectory optimization and statistical trajectory learning. We model a distribution of the trajectories

Manuscript received: Sept., 10, 2016; Revised Nov., 30, 2016; Accepted Dec., 28, 2016.

This paper was recommended for publication by Editor Nancy Amato upon evaluation of the Associate Editor and Reviewers' comments. This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No #645582 (RoMaNS).

[1]T. Osa, J. Peters, and G. Neumann are with Technische Universität Darmstadt, 64289, Germany. {osa, neumann, peters}@ias.tu-darmstadt.de

[2]A. M. Ghalamzan E. and R. Stolkin are with the University of Birmingham, UK. {a.ghalamzanesfahani, r.stolkin}@bham.ac.uk

[3]J. Peters is with Max-Planck-Institut für Intelligente Systeme, Tübingen, 72076, Germany.

demonstrated by human experts and use it to define the cost function of the optimization process. Our cost function is composed of several objectives, including smoothness, obstacles avoidance and similarity to the demonstrated distribution. We automatically tune the weight of each cost term such that the resulting trajectories match the demonstrations. A trajectory is optimized with a functional gradient as in CHOMP, and our cost function is dependent on the learned distribution. Our method is capable of handling complex environments and constraints as well as existing optimization-based methods. In addition, the trajectories planned by our method implicitly encode features of the trajectories demonstrated by human operators in a similar manner as LfD methods.

II. Related Work

Optimization-based trajectory planners have been developed for decades. One of the classical methods is called "elastic band", and was proposed by Quinlan and Khatib [7], [8]. In this method, a trajectory is represented as a elastic band and is optimized based on the potential field of the given environments. Zucker et al. proposed a covariant optimization based on functional gradient, CHOMP, and showed the effectiveness of their method on multiple robot platforms. Recent work by Marinho et al. extended the functional gradient optimization to Reproducing Kernel Hilbert Space (RKHS) [9]. Schulman et al. formulated motion planning as a sequential convex optimization and developed a motion planning framework called TrajOpt in [4]. These optimization-based methods can generate trajectories by explicitly optimizing the cost functions that represent collisions with obstacles or the smoothness of trajectories under various constraints if the model of the environment is available.

In contrast, LfD methods have been investigated in recent years as an approach for achieving intuitive robotic motion programming. Paraschos et al. developed a framework called ProMP to model the distribution of the trajectory in parameter space [10]. The study by Osa et al. showed that online trajectory planning could be achieved by learning the contextual trajectory distribution [11]. These methods model the distribution of the trajectories demonstrated by human operators and generalize the trajectories to new situations based on the learned distributions. They implicitly encode skills of human operators without using hand-coded cost/reward functions. Although robot LfD approaches can easily encode skills of human operators, a main problem arises during generalization of the demonstrated behavior in a complex environment. For instance, adapting a trajectory generated by this approach to an environment with new obstacles is still an ongoing research. A typical way to achieve obstacle avoidance is to set repulsive motions around obstacles or to diminish motions towards obstacles [12], [13]. The study by Ghalamzan et al. proposed to learn obstacle avoidance using inverse optimal control [14]. Recent work in [15] addressed a problem of avoiding a collision with a human operator based on the uncertainty of her/his motion. Although real-time obstacle avoidance is demonstrated in [12], [13], these methods consider only the collision of the end-effector and do not address the constraints of kinematics and the collision of body points other than the end-effector.

Path planning using sampling-based methods has been applied to various robotic systems as they can be used in complex environments [1], [2]. One approach to bridge the gap between the LfD methods and sampling-based methods is shown in [16]. Ye and Alterovitz extended sampling-based methods in order to take into account the distribution of the demonstrated trajectories. Although sampling-based methods can deal with complex path planning scenarios, these methods are computationally inefficient compared to optimization-based methods [3], [4]. Thus, it is desirable to bridge the gap between the optimization-based and LfD methods.

In LfD, we can design a controller that adaptively determines the controller gain based on the variance of the trajectories [10], [17]–[19]. For example, the studies by Calinon et al investigated methods for learning the trajectory distribution and controlling the system using LQR where the trajectory distribution defines the cost function [18], [19]. Intuitively, these methods control the stiffness of the robot manipulators based on the variance of the demonstrated trajectories, but they do not adapt the desired trajectory itself. Although Muhlig et al. presented variance-based movement optimization [20], they did not addressed collision avoidance. On the contrary, we optimize a trajectory for obstacle avoidance and use functional gradient optimization instead of LQR.

III. Functional Gradient Trajectory Optimization Guided by Demonstrations

In this section, we introduce our new functional gradient trajectory optimization algorithm to modify expert demonstrations due to several constraints such as obstacle avoidance and smoothness constraints. In the case of obstacle avoidance, there is a trade-off between the collision risk and human preference. In practice, the state of the obstacles is not fully observable, and it is necessary to take margins to avoid collisions with obstacles. However, deviation from the demonstrated trajectories may lead to deviation from the desired behavior. To control this trade-off between collision risk and human preference, we use the distribution of the demonstrated trajectories. We assume that a high variance of the demonstrated trajectories indicates a weak preference of the experts. Conversely, we assume that a low variance of the demonstrated trajectories indicates a strong preference of the experts. Based on this assumption, we embed the distribution of the demonstrated trajectories in trajectory optimization. The intuitive interpretation of our trajectory optimization is the following: At a phase where the variance of the demonstrated trajectory is low, the shape of the trajectory is rigid and its deformation for avoiding obstacles is minimal. Meanwhile, at a phase where the variance of the demonstrated trajectory is high, the shape of the trajectory is flexible and its deformation for avoiding the obstacle can be large enough even in the presence of measurement noise of the obstacle positions. We describe our cost functional in Section III-B Although we use a few cost terms in our cost functional, the weights on each cost term can be tuned by matching the demonstrated trajectories as described in Section III-D.

A. Modeling the Expert's Trajectory Distribution

We assume that the trajectory $\boldsymbol{\tau}$ is given by a sequence of the states of the system $\boldsymbol{x}$ as $\boldsymbol{\tau} = [\boldsymbol{x}(0), \cdots, \boldsymbol{x}(N)]$ where N is the number of the time steps of the trajectory. We model the distribution of the state at time t as a Gaussian distribution

$$p\left(\boldsymbol{x}(t)\right) \sim \mathcal{N}(\boldsymbol{\mu}(t), \boldsymbol{\Sigma}(t)). \tag{1}$$

where $\boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma}_t$ are the mean and the covariance of the state at time t. Therefore, the distribution of the trajectory $\boldsymbol{\tau}$ is given by

$$p(\boldsymbol{\tau}) = \prod_{t=0}^{N} p\left(\boldsymbol{x}(t)\right) = \prod_{t=0}^{N} \mathcal{N}(\boldsymbol{\mu}(t), \boldsymbol{\Sigma}(t)). \tag{2}$$

We assume that the demonstrations are performed under various contexts $\boldsymbol{s}_i$. For example, the context could be given by the target location of an object to grasp. We can now model the conditional distribution of the demonstrated trajectory given the context and generalize the demonstrated trajectories to new scenes. The use of conditional distributions of the demonstrated trajectory in our trajectory optimization is straightforward. In this paper, we use locally weighted estimation to model such a conditional distribution [21], [22], however, our approach is not limited to specific regression methods and other regression methods such as Gaussian Mixture Regression and Gaussian Process Regression could be used instead.

We assume that we have a dataset of M demonstrations $\mathcal{D} = \{\boldsymbol{\tau}^i, \boldsymbol{s}^i\}_{i=1}^M$ where $\boldsymbol{s}$ is a vector that represents the context of the task. Since the execution speed may be different between demonstrations, the time alignment of the demonstrated trajectories needs to synchronized. Therefore, we synchronize the demonstrated trajectories in time domain using Dynamic Time Warping [23] in the same manner as in [24]. Given a test context $\boldsymbol{s}_{\text{test}}$, the locality weight of the ith sample can be computed as

$$w_i = \exp\left(-\frac{(\boldsymbol{s}^i - \boldsymbol{s}_{\text{test}})^\top (\boldsymbol{s}^i - \boldsymbol{s}_{\text{test}})}{h}\right), \tag{3}$$

where h is a constant that determines the bandwidth of the squared exponential kernel. The joint distribution of the context and the state is given by a Gaussian distribution

$$p(\boldsymbol{x}(t), \boldsymbol{s}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_{\boldsymbol{x}}(t) \\ \boldsymbol{\mu}_{\boldsymbol{s}} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{x}}(t) & \boldsymbol{\Sigma}_{\boldsymbol{xs}}(t) \\ \boldsymbol{\Sigma}_{\boldsymbol{xs}}(t)^\top & \boldsymbol{\Sigma}_{\boldsymbol{s}} \end{bmatrix}\right) \tag{4}$$

where the weighted mean and variance for the tth time step are given by

$$\boldsymbol{\mu}_{\boldsymbol{x}}(t) = \frac{\sum_{i=1}^M w_i \boldsymbol{x}^i(t)}{\sum_{i=1}^M w_i}, \ \boldsymbol{\mu}_{\boldsymbol{s}} = \frac{\sum_{i=1}^M w_i \boldsymbol{s}^i}{\sum_{i=1}^M w_i} \tag{5}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{x}}(t) = \frac{\sum_{i=1}^M w_i (\boldsymbol{x}^i(t) - \boldsymbol{\mu}_{\boldsymbol{x}})(\boldsymbol{x}^i(t) - \boldsymbol{\mu}_{\boldsymbol{x}})^\top}{\sum_{i=1}^M w_i}, \tag{6}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{s}} = \frac{\sum_{i=1}^M w_i (\boldsymbol{s}^i - \boldsymbol{\mu}_{\boldsymbol{s}})(\boldsymbol{s}^i - \boldsymbol{\mu}_{\boldsymbol{s}})^\top}{\sum_{i=1}^M w_i}, \tag{7}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{xs}}(t) = \frac{\sum_{i=1}^M w_i (\boldsymbol{x}^i(t) - \boldsymbol{\mu}_{\boldsymbol{x}})(\boldsymbol{s}^i - \boldsymbol{\mu}_{\boldsymbol{s}})^\top}{\sum_{i=1}^M w_i}. \tag{8}$$

Using the weighted mean and variance, the conditional expectation and variance of the state $\boldsymbol{x}(t)$ given the context $\boldsymbol{s}_{\text{test}}$ can be computed as

$$\begin{aligned} \mathbb{E}[\boldsymbol{x}(t)|\boldsymbol{s}_{\text{test}}] &= \boldsymbol{\mu}_{\boldsymbol{x}} + \boldsymbol{\Sigma}_{\boldsymbol{x}_t \boldsymbol{s}} \boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1} (\boldsymbol{s}_{\text{test}} - \boldsymbol{\mu}_{\boldsymbol{s}}), \\ \boldsymbol{\Sigma}_{\boldsymbol{x}|\boldsymbol{s}_{\text{test}}}(t) &= \boldsymbol{\Sigma}_{\boldsymbol{x}} - \boldsymbol{\Sigma}_{\boldsymbol{xs}} \boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1} \boldsymbol{\Sigma}_{\boldsymbol{sx}} \end{aligned} \tag{9}$$

We can obtain the conditional distribution of the trajectory given the context by computing (9) for $t = 0, \cdots, N$.

B. Demonstrated-Guided Cost Function

Let $\boldsymbol{\tau}^c$ denote the trajectory given by the sequence of the robot configurations as $\boldsymbol{\tau}^c = [\boldsymbol{q}(0), \ldots, \boldsymbol{q}(N)]$. The trajectory $\boldsymbol{\tau}^c$ can be considered as a function that maps time $t \in [0, 1]$ to a configuration $\boldsymbol{q} \in \mathbb{R}^D$. At the same time, the cost function can be represented as a function of the trajectory $\boldsymbol{\tau}^c$. Therefore, the cost function can be considered as a function of a function, which is referred to as a functional as described in [3], [7], [8]. In our approach, we minimize the cost functional

$$\begin{aligned} \mathcal{C}(\boldsymbol{\tau}^c) = \lambda_{\text{smooth}} c_{\text{smooth}}(\boldsymbol{\tau}^c) + \lambda_{\text{obs}} c_{\text{obs}}(\boldsymbol{\tau}^c) \\ + \lambda_{\text{demo}} c_{\text{demo}}(\boldsymbol{\tau}^c) + \lambda_{\text{man}} c_{\text{man}}(\boldsymbol{\tau}^c), \end{aligned} \tag{10}$$

where $c_{\text{smooth}}(\boldsymbol{\tau}^c)$ is the cost that induces the smoothness, $c_{\text{obs}}(\boldsymbol{\tau}^c)$ is the term that penalizes the collision with obstacles, $c_{\text{demo}}(\boldsymbol{\tau}^c)$ is the term that penalizes the deviation from the mean of the demonstrated trajectories, and $c_{\text{man}}(\boldsymbol{\tau}^c)$ is the term that induces the manipulability of a robotic manipulator. $[\lambda_{\text{smooth}}, \lambda_{\text{obs}}, \lambda_{\text{demo}}, \lambda_{\text{man}}]$ are the weights on the cost terms.

1) Matching the Demonstrations: In our cost functional, the term $c_{\text{demo}}(\boldsymbol{\tau}^c)$ penalizes the deviation of the end effector trajectory from the demonstrated end-effector trajectory. As described in the previous section, we model the distribution of the end-effector trajectories demonstrated by experts as

$$p(\boldsymbol{x}_{\text{e}}(t)) \sim \mathcal{N}\left(\boldsymbol{\mu}_{\text{e}}(t), \boldsymbol{\Sigma}_{\text{e}}(t)\right), \tag{11}$$

where $\boldsymbol{x}_{\text{e}}(t)$ is the position of the end effector at time t. We define $c_{\text{demo}}(\boldsymbol{\tau}^c)$ as

$$c_{\text{demo}}(\boldsymbol{\tau}^c) = \frac{1}{2} \int \tilde{\boldsymbol{x}}_{\text{e}}^\top \boldsymbol{\Sigma}_{\text{e}}^{-1} \tilde{\boldsymbol{x}}_{\text{e}} dt, \tag{12}$$

where $\tilde{\boldsymbol{x}}_{\text{e}}$ is given by $\tilde{\boldsymbol{x}}_{\text{e}} = \boldsymbol{x}_{\text{e}}(t) - \boldsymbol{\mu}_{\text{e}}(t)$. By using this cost term $c_{\text{demo}}(\boldsymbol{\tau}^c)$, the cost of the deviation from the demonstration is high at a phase when the variance of the demonstrated trajectories is low. Conversely, at a phase when the variance of the demonstrated trajectory is high, the cost of the deviation from the demonstration is low. Similar cost functions appear in [17]–[20], although these previous studies did not address collision avoidance. The functional gradient of $c_{\text{demo}}(\boldsymbol{\tau}^c)$ is given by

$$\bar{\nabla} c_{\text{demo}}(\boldsymbol{\tau}^c) = \boldsymbol{J}^\top \boldsymbol{\Sigma}_{\text{e}}^{-1} \tilde{\boldsymbol{x}}_{\text{e}}, \tag{13}$$

where $\boldsymbol{J}$ is the Jacobian of the manipulator.

2) Obstacle Avoidance: We additionally exploit the demonstrated distribution in the cost functional associated with the obstacle. In order to evaluate the obstacle cost of the whole body of the manipulator, we consider body points of the given manipulator. To formulate the obstacle cost, we use the distribution of the body points in demonstrated trajectories in task space

$$p\left(\boldsymbol{x}_u(t)\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_u(t), \boldsymbol{\Sigma}_u(t)\right), \tag{14}$$

where u is the index of the body point and $\boldsymbol{x}_u(t)$ is the position of the body point u in task space at time t. We define the obstacle cost $c_{\text{obs}}(\boldsymbol{\tau})$ as

$$c_{\text{obs}}(\boldsymbol{\tau}) = \frac{1}{2}\int_0^1 \int_{\mathcal{B}} c\left(\boldsymbol{x}_u(t)\right) \left\| \frac{d}{dt}\boldsymbol{x}_u(t) \right\| du\ dt, \tag{15}$$

where $\mathcal{B}$ is a set of body points which comprise the robot body, and the local collision cost function $c(\boldsymbol{x}_u)$ is defined as

$$c(\boldsymbol{x}_u) = \begin{cases} 0 & \text{if} \quad d(\boldsymbol{x}_u) > \sigma_u^{\max}\epsilon, \\ \frac{1}{2\sigma_u^{\max}\epsilon}(d(\boldsymbol{x}_u) - \sigma_u^{\max}\epsilon)^2 & \text{if} \quad 0 < d(\boldsymbol{x}_u) < \sigma_u^{\max}\epsilon, \\ d(\boldsymbol{x}_u) + \frac{1}{2}\sigma_u^{\max}\epsilon, & \text{if} \quad d(\boldsymbol{x}_u) < 0, \end{cases} \tag{16}$$

The term $\sigma_u^{\max}$ is defined as the maximum diagonal element of the covariance matrix $\boldsymbol{\Sigma}_u$, ϵ is a constant that scales the margin, and $d(\boldsymbol{x}_u)$ represents a signed distance between the body point u and the nearest obstacle. $d(\boldsymbol{x}_u)$ is negative when the body point is inside obstacles, and zero at the boundary. In this formulation of the collision cost, the margin for the collision avoidance is proportional to the variance of the demonstrated trajectories. In our formulation, the variance of the demonstrated trajectories is used to control the trade-off between the collision risk and the human preference The functional gradient of $c_{\text{obs}}(\boldsymbol{\tau}^c)$ is given by

$$\bar{\nabla} c_{\text{obs}}(\boldsymbol{\tau}^c) = \int_{\mathcal{B}} \boldsymbol{J}^\top \left\|\dot{\boldsymbol{x}}_u\right\| \left[\left(\boldsymbol{I} - \frac{\dot{\boldsymbol{x}}_u \dot{\boldsymbol{x}}_u^\top}{\left\|\dot{\boldsymbol{x}}_u\right\|^2} \right) \nabla c - c\kappa \right] du, \tag{17}$$

where κ is defined as

$$\kappa = \left(\boldsymbol{I} - \frac{\dot{\boldsymbol{x}}_u \dot{\boldsymbol{x}}_u^\top}{\left\|\dot{\boldsymbol{x}}_u\right\|^2} \right) \frac{\ddot{\boldsymbol{x}}_u}{\left\|\dot{\boldsymbol{x}}_u\right\|^2}\ . \tag{18}$$

3) Smoothness: As in CHOMP, the smoothness cost $c_{\text{smooth}}(\boldsymbol{\tau})$ is defined as

$$c_{\text{smooth}}(\boldsymbol{\tau}^c) = \frac{1}{2}\int \left\| \frac{d}{dt}\boldsymbol{\tau}^c(t) \right\|^2 dt. \tag{19}$$

Since we evaluate the smoothness of a discrete-time trajectory $\boldsymbol{\tau}^c \in \mathbb{R}^{T\times d}$, the smoothness cost can be computed as

$$c_{\text{smooth}}(\boldsymbol{\tau}^c) = \frac{1}{2}\left\| K\boldsymbol{\tau}^c + \boldsymbol{e} \right\|^2, \tag{20}$$

where the finite differentiation matrix $\boldsymbol{K} \in \mathbb{R}^{N\times N}$ and the vector $\boldsymbol{e} \in \mathbb{R}^{N\times D}$ are given by

$$\boldsymbol{K} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ -1 & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & \dots & -1 & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & -1 & 0 \end{bmatrix}, \tag{21}$$

$$\boldsymbol{e} = \left[-\boldsymbol{q}_1, 0, \dots, 0, \boldsymbol{q}_{N-1}\right]^\top. \tag{22}$$

By using this representation, the functional gradient of $c_{\text{smooth}}(\boldsymbol{\tau}^c)$ is given by

$$\bar{\nabla} c_{\text{smooth}}(\boldsymbol{\tau}^c) = \boldsymbol{K}^\top(\boldsymbol{K}\boldsymbol{\tau}^c + \boldsymbol{e}). \tag{23}$$

4) Manipulability: The manipulability is often used as a metric of the capability of robotic manipulators in a given configuration. By maximizing the manipulability, the trajectory optimization finds a solution avoiding the singularity. We use the cost functional associated with the manipulability defined as

$$c_{\text{man}}(\boldsymbol{\tau}^c) = -\int \sqrt{\det\left(\boldsymbol{J}^\top(\boldsymbol{\tau}^c(t))\boldsymbol{J}(\boldsymbol{\tau}^c(t))\right)} dt. \tag{24}$$

The gradient of the manipulability cost can be computed numerically. In order to maximize the manipulability while maintaining the end effector close to the demonstrated trajectories, we project the gradient of the manipulability onto the null space of the manipulator as $(\boldsymbol{I} - \boldsymbol{J}^\dagger\boldsymbol{J})\bar{\nabla} c_{\text{man}}(\boldsymbol{\tau}^c)$ where $J^\dagger$ is the pseudo-inverse of J.

C. Optimization of the Cost Functional

Based on the above discussion, the gradient of the total cost functional is given by

$$\begin{aligned} \bar{\nabla}\mathcal{C}(\boldsymbol{\tau}^c) = {} & \lambda_{\text{smooth}}\bar{\nabla} c_{\text{smooth}}(\boldsymbol{\tau}^c) + \lambda_{\text{obs}}\bar{\nabla} c_{\text{obs}}(\boldsymbol{\tau}^c) \\ & + \lambda_{\text{demo}}\bar{\nabla} c_{\text{demo}}(\boldsymbol{\tau}^c) + \lambda_{\text{man}}(\boldsymbol{I} - \boldsymbol{J}^\dagger\boldsymbol{J})\bar{\nabla} c_{\text{man}}(\boldsymbol{\tau}^c). \end{aligned} \tag{25}$$

To optimize the total cost functional $\mathcal{C}(\boldsymbol{\tau}^c)$, we linearize the cost functional as

$$\mathcal{C}(\boldsymbol{\tau}^c) \approx \mathcal{C}(\boldsymbol{\tau}_i^c) + (\boldsymbol{\tau}^c - \boldsymbol{\tau}_i^c)^\top \nabla\mathcal{C}(\boldsymbol{\tau}_i^c) \tag{26}$$

and optimize the optimization problem

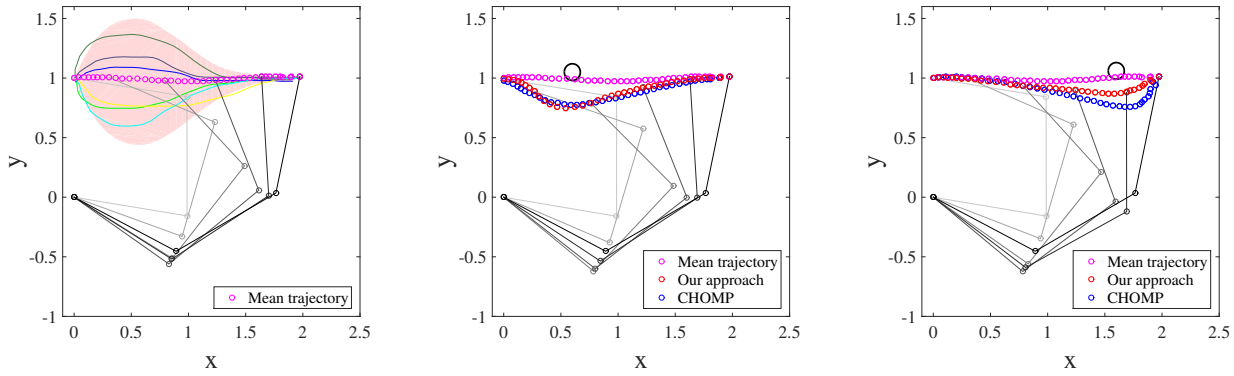
$$\boldsymbol{\tau}_{i+1}^c = \arg\min_{\boldsymbol{\tau}^c} \left\{ \mathcal{C} + (\boldsymbol{\tau}^c - \boldsymbol{\tau}_i^c)^\top \nabla\mathcal{C} + \frac{\eta}{2}\left\|\boldsymbol{\tau}^c - \boldsymbol{\tau}_i^c\right\|_M \right\}, \tag{27}$$

where $\left\|\boldsymbol{\tau}^c - \boldsymbol{\tau}_i^c\right\|_M$ denotes the distance between $\boldsymbol{\tau}^c$ and $\boldsymbol{\tau}_i^c$ with respect to the Riemannian metric $\boldsymbol{M}$ given by $(\boldsymbol{\tau}^c - \boldsymbol{\tau}_i^c)^\top \boldsymbol{M}(\boldsymbol{\tau}^c - \boldsymbol{\tau}_i^c)$. The last term in (27) behaves as a regularization term in the optimization and penalizes a large change in the trajectory update. If we set the metric $\boldsymbol{M} = \boldsymbol{K}^\top\boldsymbol{K}$ as in CHOMP [3], the functional gradient will be uniformly propagated to the whole trajectory. In contrast, we use the Riemannian metric

$$\boldsymbol{M} = \boldsymbol{K}^\top\boldsymbol{W}\boldsymbol{K} \tag{28}$$

where $\boldsymbol{W}$ is a diagonal matrix and its diagonal elements are given by

$$W_{tt} = k_{w1}\exp(-k_{w2}\sigma_{\text{e}}^{\max}(t)), \tag{29}$$



(a) (b) (c)

Fig. 2: The behavior of the proposed trajectory planner. Black circles represent the obstacles. (a) The distribution of the demonstrated trajectories and the motions of the three-link manipulator. (b) At time steps where the variance of the demonstrated trajectories is large, the trajectory is adapted flexibly. (c) At time steps where the variance of the demonstrated trajectories is small, our approach keeps to stay close to the mean trajectory of the demonstrations.

k_{w1} and k_{w2} are constant and $\sigma_{\mathrm{e}}^{\max}(t)$ is the maximum value of the diagonal elements of $\Sigma_{\mathrm{e}}(t)$. By using this Riemannian metric, we can penalize large changes of the trajectory at time steps where the variance is small in the demonstrations. Based on the linearization as in (27), the trajectory update rule is given by

$$\boldsymbol{\tau}_{i+1}^{c} = \boldsymbol{\tau}_{i}^{c} - \eta_i \boldsymbol{M}^{-1} \bar{\nabla} \mathcal{C}(\boldsymbol{\tau}^{c}), \tag{30}$$

where η_i is the step-size. Since we use the Riemanian metric in (28), the functional gradient is propagated to the trajectory based on the variance of the demonstrated trajectories.

D. Automatic Tuning of Weight Parameters

In our formulation of the cost function, we need to tune the weight parameters $\boldsymbol{\theta} = [\lambda_{\mathrm{smooth}}, \lambda_{\mathrm{obs}}, \lambda_{\mathrm{demo}}, \lambda_{\mathrm{man}}]$. By matching the demonstrated trajectories, these parameters can be tuned without heuristic parameter selection.

In this procedure, human experts demonstrate how to avoid obstacles, and we record the trajectory $\boldsymbol{\tau}_{\mathrm{demo}}$ and the condition of obstacle $\boldsymbol{s}_{\mathrm{obs}}$. We can now obtain weight parameters such that the optimized trajectories reproduce the collision avoidance behavior of the experts. For this purpose, we use the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) algorithm, which is a well-known stochastic optimization method [25]. We used CMA-ES to optimize the objective function

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{M'} \int \left\| \boldsymbol{\tau}_{\boldsymbol{\theta}}^{i}(t) - \boldsymbol{\tau}_{\mathrm{demo}}^{i}(t) \right\| dt, \tag{31}$$

where M' is the number of demonstrations, $\boldsymbol{\tau}_{\mathrm{demo}}^{i}$ is the trajectory demonstrated by the expert under the ith obstacle condition, and $\boldsymbol{\tau}_{\boldsymbol{\theta}}^{i}$ is the optimized trajectory using our method under the same condition using the parameter vector $\boldsymbol{\theta}$. In our experiments, we fixed one of the parameters and tuned the rest of the parameters with CMA-ES. Algorithm 1 summarizes the procedure for automatic parameter tuning. Although we used CMA-ES to optimize the weight parameters, any other optimization method can be used.

Algorithm 1 Parameter Tuning by Learning from Demonstration

1: Collect the demonstration of avoiding obstacles $\mathcal{D}^{\mathrm{avoid}} = \{\boldsymbol{\tau}_{\mathrm{demo}}^{i}, \boldsymbol{s}_{\mathrm{obs}}^{i}\}_{i=1}^{M'}$

2: Run optimization, e.g. CMA-ES, to tune the weights $\boldsymbol{\theta}$

$$\boldsymbol{\theta} = \arg\min \sum_{i=1}^{M'} \int \left\| \boldsymbol{\tau}_{\boldsymbol{\theta}}^{i}(t) - \boldsymbol{\tau}_{\mathrm{demo}}^{i}(t) \right\| dt$$

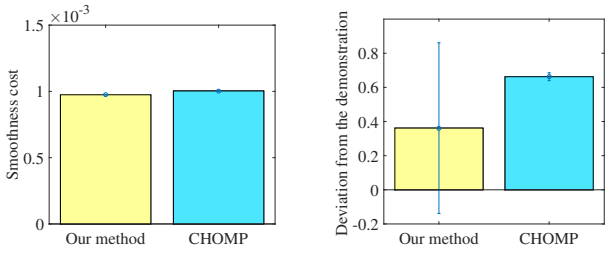
IV. Experiments

We illustrate the behavior of our motion planning algorithm for a simulated three-link manipulator in 2D space at first. Thereafter, we will show the benefit of guiding the trajectory optimization by demonstrations in real robot experiments.

A. Simulated 3-Link Manipulator

We evaluated the performance of our approach with a three-link manipulator in 2D space and compared our approach with CHOMP. In this simulation, a point-to-point motion is learned. Ten trajectories were given to the system as demonstrations, and their distribution was modeled as described in Section III. The distribution of the demonstrated trajectories is visualized in Fig. 2 (a). To illustrate the behavior of our trajectory optimization, we choose the demonstrations such that the variance of the trajectories is high in the beginning of the motion while it is much smaller in the end of the motion. The weight parameters were tuned by matching the demonstrations as described in Section III-D. For tuning the weight parameters, four trajectories were demonstrated in scenes with different obstacles positions. The parameters were randomly initialized, and the result of the parameter tuning was $\boldsymbol{\theta} = [3.9, 0.15, 0.85, 0.15]$. Our method and CHOMP were initialized with the mean trajectory of the demonstrated trajectories.

The results show that our approach adaptively controls the flexibility of the trajectory in the trajectory optimization.



(a) (b)

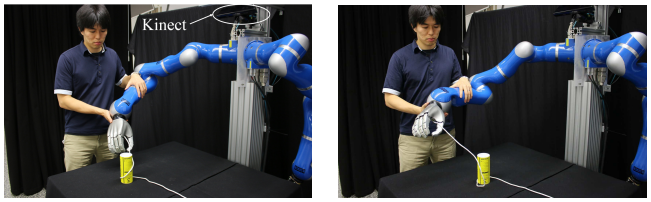
Fig. 3: Comparison with CHOMP. (a)The smoothness cost of the trajectory resulting from our method is lower than the result of CHOMP. (b)The variance of the deviation from the demonstrated trajectories is substantially large in our method. This results indicates that our method adaptively controls the deformation of trajectories.

As shown in Fig. 2 (a), the variance of the demonstrated distribution is high in the beginning of the motion, and the variance is low in the end of the motion. To demonstrate the behavior of our approach, we show the two scenes in Fig. 2 (b) and (c). In Fig. 2(b), the obstacle was placed at the beginning of the trajectory where the variance is high. In Fig. 2(c), the obstacle was placed at the end of the trajectory, i.e, with small trajectory variance. As expected, the deformation of the trajectory resulting from our method is large in Fig. 2 (b) while it is much smaller in Fig. 2 (c). In contrast, the trajectories obtained from CHOMP showed a similar behavior for both scenarios as the algorithm is ignorant to the variance of the demonstrated distribution.

To evaluate the quality of the planned trajectories, we randomly placed an obstacle along the trajectory and computed the sum of the norm of the acceleration over the trajectory $\sum \left\| \frac{d^2}{dt^2} \boldsymbol{\tau}(t) \right\|^2$, and the deviation from the mean demonstrated trajectory $\sum \|\boldsymbol{\tau}(t) - \mathbb{E}_{\text{demo}}[\boldsymbol{\tau}(t)]\|^2$. The results are shown in Fig. 3. The smoothness of the trajectories is comparable between our method and CHOMP as shown in Fig. 3 (a). The variance of the deviation from the demonstrated trajectories is substantially larger in our method compared to CHOMP. This result indicates that our method adaptively control the deformation of the trajectories, while the deformation is uniform over the trajectory in CHOMP.

B. Experiments with a Real Robot

To evaluate the performance of our method in 3D space, we performed experiments with a robot manipulator with 7 DOF. In this experiment, a motion to disentangle a rope from an object is learned. The task setup is shown in Fig. 4. The rope is initially winded around a yellow can. The system learned how to disentangle the rope from the yellow can from demonstrations while using trajectory optimization to avoid newly introduced obstacles. This motion involves performing a loop around the entangled object, which is hard to represented as a simple point-to-point motion. The demonstrations were performed eight times with different positions of the entangled object. Obstacles were not presented in the demonstrations.



(a) (b)

Fig. 4: A human demonstrated the disentangling motion by kinesthetic teaching. The position of the object is measured by a Kinect. The demonstrations were performed in scenes without obstacles.

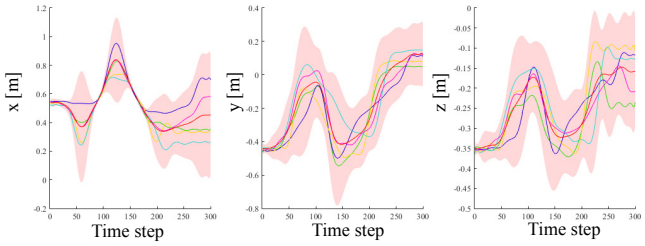


Fig. 5: Distribution of the demonstrated trajectories of the end effector in the real robot experiment.

A human operator demonstrated the disentangling motion by kinesthetic teaching as shown in Fig. 4. The context $\boldsymbol{s}$ of the task is defined

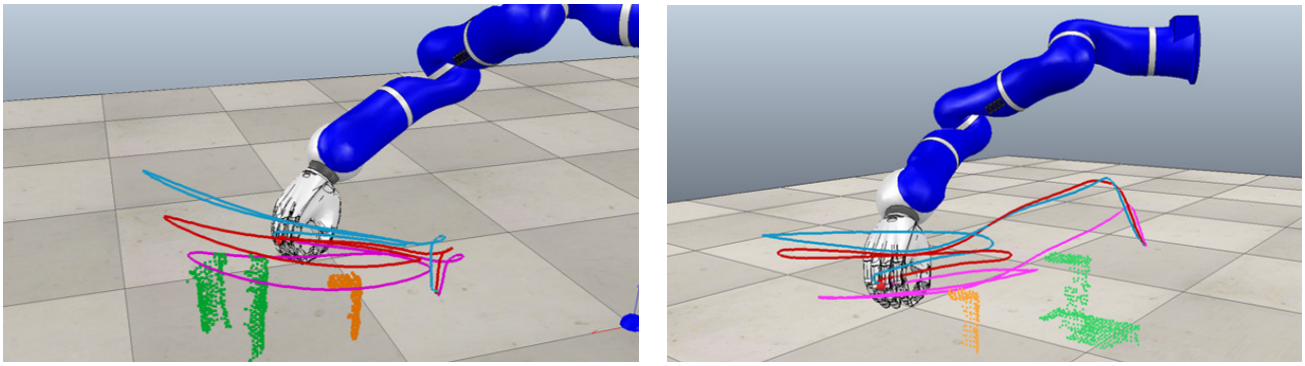
$$\boldsymbol{s} = [x_{\text{target}}, y_{\text{target}}, z_{\text{target}}], \tag{32}$$

where $[x_{\text{target}}, y_{\text{target}}, z_{\text{target}}]$ represents the position of the object.

By modeling the conditional distribution of the trajectories as described in Section III-A, the motion can be generalized to new contexts. The demonstrated trajectories and their distribution is shown in Fig. 5. An example of the conditional distribution of the trajectory given a new context is shown in Fig. 6. In the predicted distribution, the variance of the trajectory is large in the end of the motion, and the variance is relatively small in the middle of the motion. Considering the task, this result is intuitive as the looping motion for disentangling the object needs to be consistent, while the end motion is not critical to the success of the task.

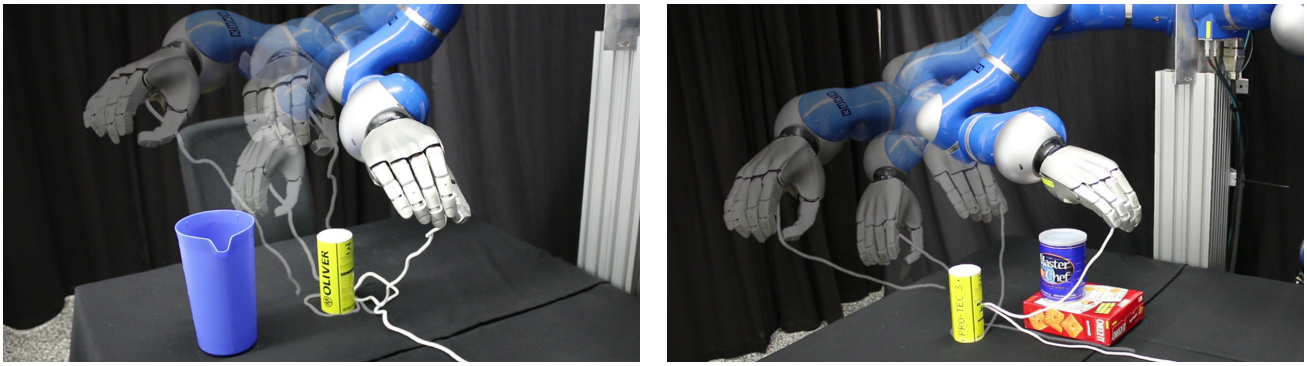
To evaluate our trajectory optimization algorithm, we placed obstacles in the scenes as shown in Fig. 7. In the scene shown in Fig. 7 (a), we placed an obstacle such that the trajectory encounters the obstacle at the middle of the motion where the variance of the trajectory distribution is low. In the scene shown in Fig. 7 (b), we placed obstacles such that the trajectory encounters the obstacles at the end of the motion where the variance of the trajectory distribution is high. The shape and position of the obstacles were captured as point clouds using a Kinect. To compute the collision costs efficiently, the obstacles were represented as set of spheres. In this experiment, we heuristically chose the weight parameters.

The comparison between our approach and CHOMP is illustrated in Fig. 8. The three lines in Fig. 8 show the trajectory of the middle finger of the robot. The red line represents the trajectory obtained from our approach, the blue line represents the trajectory obtained from CHOMP, and the magenta line represents the initial trajectory predicted as a



(a) (b)

Fig. 8: Comparison of the resulting trajectories. The red line represents the trajectory obtained from our approach, and the blue line represents the trajectory obtained from CHOMP. The magenta line represents the initial trajectory predicted as a conditional expectation from the demonstrated trajectories. Scenes in (a) and (b) correspond to the scenes shown in Fig. 7 (a) and (b), respectively. The green dots represent the point cloud of the obstacles, and the orange dots represent the point cloud of the yellow can in Fig. 7. The trajectory obtained from CHOMP largely deviated from the demonstrated trajectory. In contrast, the trajectory obtained from our method stays close to the demonstrated trajectory.



(a) (b)

Fig. 9: Motion of the manipulator executing the trajectory obtained from our approach. Scenes in (a) and (b) correspond to the scenes shown in Fig. 8(a) and (b), respectively.

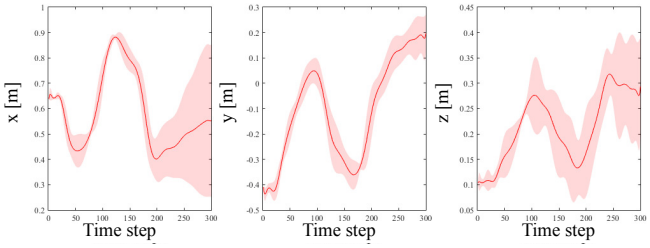
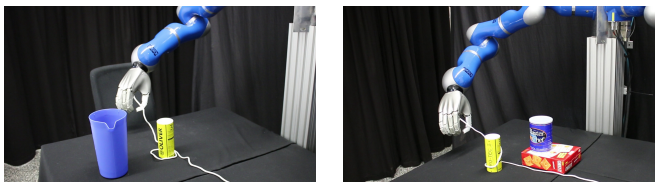


Fig. 6: Distribution of the trajectory predicted for the context shown in Fig. 8(a).

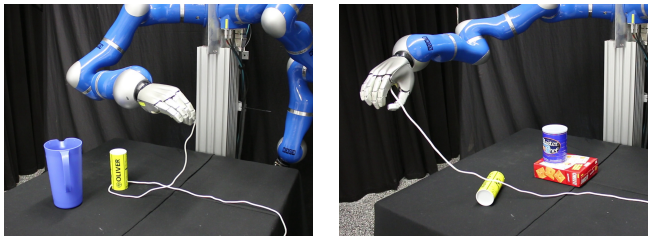


(a) (b)

Fig. 7: Initial state of the disentangling motion. A rope is entangled with the yellow can. We placed obstacles in different positions to evaluate the behavior of our trajectory optimization. In Fig. (a), the obstacle was placed to perturb the trajectory at the middle of the motion. In Fig. (b), the obstacle was placed to perturb the trajectory at end of the motion.

conditional expectation from the demonstrated trajectories. In the scene shown in Fig. 8(a), where the obstacle was located to perturb the looping motion, the trajectory from CHOMP largely deviated from the predicted trajectory. In Fig. 8(b), the trajectory generated by CHOMP largely deviated from the predicted trajectories even in the phase which is far from the obstacle. In contrast, the trajectory obtained from our approach was close to the predicted trajectory in the middle of the motion, while it largely deviated from the demonstration to avoid obstacles only when the obstacle is close. On the other hand, our approach generated the trajectory close to the demonstrated trajectories. These results demonstrate that, in our method, the trajectory is deformed flexibly in the phase where the prediction variance is high and that the trajectory stays close to the predicted trajectory when the prediction variance is low. This result shows the advantage of our approach against CHOMP. In CHOMP, the margin to avoid the obstacles is constant over the trajectory while our approach adaptively controls the margin to avoid the obstacle based on the distribution of the demonstrated trajectories.

Fig. 9 shows the real robot executing the trajectory obtained



(a) (b)

Fig. 10: The trajectory obtained from CHOMP failed to perform the task properly in our experiments as the trajectory obtained from CHOMP largely deviated from the demonstrated behavior. (a) The rope was not totally disentangled in the scene shown in Fig. 7 (a). (b) The yellow can fell down during the task in the scene shown in Fig. 7 (b).

from our approach. The robot could disentangle the robe from the object while avoiding obstacles. However, the task was not properly performed when the robot executed the trajectory obtained from CHOMP as shown in Fig. 10 since the trajectory obtained from CHOMP largely deviates from the demonstrated behavior. For example, in the scene shown in Fig. 10 (a), the rope was not properly disentangled and in Fig. 10 (a) the object entangled with the rope fell over during the motion. Our experimental results indicate that the demonstrated trajectories implicitly encode the demonstrator's behavior, and large deviations from the demonstrated trajectories may cause unexpected results.

V. Discussion and Conclusions

The benefit of our approach is that the resulting trajectory implicitly encodes the important components of the demonstrated behavior. On the contrary, CHOMP only considers the start and goal positions and collision avoidance as objectives. However, in tasks such as the disentangle-a-rope task, the topological shape of the entire trajectory is more important than the start and goal positions. Hence, CHOMP failed on the disentangle-a-rope task. Our approach attempts to match the demonstrated behavior during the entire trajectory where the demonstrated variance profile is used to adapt the flexibility of the deformation.

We proposed a motion-planning framework with functional gradient optimization that incorporates human demonstrations. Our approach bridges the gap between LfD approaches and optimization-based approaches for motion planning with collision avoidance. By leveraging the distribution of demonstrated trajectories for trajectory optimization, we can obtain a collision-free trajectory that matches the behavior demonstrated by human experts. Through experiments with simulations and a real robotic system, we validated that our approach optimizes the trajectory to avoid obstacles and encodes the demonstrated behavior in the resulting trajectory.

References

[1] L. Kavraki and J. Latombe, *Probabilistic roadmaps for robot path planning*. John Wiley, 1998, pp. 33–53.

[2] S. LaValle and J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," *Algorithmic and Computational Robotics: New Directions*, pp. 293–308, 2001.

[3] M. Zucker, N. Ratliff, A. Dragan, M. Pivtoraiko, M. Klingensmith, C. Dellin, J. A. Bagnell, and S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, pp. 1164–1193, 2013.

[4] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, 2014.

[5] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Springer Handbook of Robotics*. Springer Berlin Heidelberg, 2008, ch. Robot Programming by Demonstration, pp. 1371–1394.

[6] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[7] S. Quinlan and O. Khatib, "Elastic bands: connecting path planning and control," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 2, 1993, pp. 802–807.

[8] S. Quinlan, "Real-time modification of collision-free paths," Ph.D. dissertation, Stanford University, 1994.

[9] Z. Marinho, B. Boots, A. Dragan, A. Byravan, G. J. Gordon, and S. Srinivasa, "Functional gradient motion planning in reproducing kernel hilbert spaces," in *Proceedings of Robotics: Science and Systems (R:SS)*, June 2016.

[10] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Probabilistic movement primitives," in *Proceedings of Advances in Neural Information Processing Systems 26*, 2013.

[11] T. Osa, N. Sugita, and M. Mitsuishi, "Online trajectory planning in dynamic environments for surgical task automation," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[12] H. Hoffmann, P. Pastor, D. H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[13] S. M. Khansari-Zadeh and A. Billard, "A dynamical system approach to realtime obstacle avoidance," *Autonomous Robots*, vol. 32, no. 4, pp. 433–454, 2012.

[14] A. M. Ghalamzan, C. P. E., G. D. Hager, and L. Bascetta, "An incremental approach to learning generalizable robot tasks from human demonstration," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 5616–5621.

[15] M. Zeestraten, A. Pereira, M. Althoff, and S. Calinon, "Online motion synthesis with minimal intervention control and formal safety guarantees," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016.

[16] G. Ye and R. Alterovitz, "Demonstration-guided motion planning," in *Proceedings of International Symposium on Robotics Research (ISRR)*, 2011.

[17] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenéz, and C. Torras, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 513–527, 2016.

[18] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent Service Robotics (Springer)*, vol. 9, no. 1, pp. 1–29, 2016.

[19] D. Bruno, S. Calinon, and D. G. Caldwell, "Learning autonomous behaviours for the body of a flexible surgical robot," *Autonomous Robots*, pp. 1–15, 2016.

[20] M. Muhlig, M. Gienger, S. Hellbach, J. J. Steil, and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2009, pp. 1177–1184.

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

[22] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning for control," *Artificial Intelligence Review*, vol. 11, no. 1, pp. 75–113, 1997.

[23] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

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

[25] N. Hansen, "The CMA evolution strategy: a comparing review," in *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*. Springer, 2006, pp. 75–102.