IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 3, 2018, accepted September 28, 2018, date of publication October 4, 2018, date of current version October 29, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2873718

Reactive Execution of Learned Tasks With Real-Time Collision Avoidance in a Dynamic Environment

GANGFENG LIU, (Member, IEEE), CAIWEI SONG, XIZHE ZANG, (Member, IEEE), AND JIE ZHAO, (Member, IEEE)

School of Mechatronics Engineering, Harbin Institute of Technology, Harbin 150001, China

Corresponding authors: Caiwei Song (16B908052@stu.hit.edu.cn) and Xizhe Zang (zangxizhe@hit.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 91648201, Grant 51521003, Grant 61803126, and Grant U1713201, and in part by the National Key R&D Program of China under Grant 2017YFB1302301 and Grant 2017YFC0806501.

ABSTRACT This paper addresses the problem of learning from demonstration (LfD) and subsequent robot safety control in an unstructured dynamic environment different from the demonstrations. Generally, LfD has been successfully exploited for task programming, but the existing methods have not solved the problem of allowing the entire arm to avoid obstacles while satisfying the task motion constraints (e.g., the robotic arm approaching the target object while avoiding obstacles moving within the environment). To achieve this, we present an incremental LfD approach that combines a task-parameterized probabilistic model and the robot security domain to control a robot's behavior during task execution. Specifically, we propose a safety-oriented and task-oriented control strategy for redundant manipulators that makes full use of the motion redundancy of the manipulator and the space with no task restraints to satisfy the task constraints for human-robot coexistence. We then demonstrate the effectiveness of the proposed approach through a series of pick-and-pour experiments performed by a manipulator with 7 degree of freedom in a dynamic environment, where the robot must both avoid obstacles and satisfactorily complete the learned task with constraints.

INDEX TERMS Motion and path planning, moving obstacles, probabilistic modelling, robot learning from demonstration.

I. INTRODUCTION

Robots have the potential to assist humans by performing a variety of daily housework and production tasks in houses or workplaces. For example, a robot can help with housework and cooperate with workshop employees to complete assembly work. However, in most cases, robots can handle repetitive operations only by executing pre-defined programs. The key challenge for robots in completing tasks autonomously is to ensure human safety while planning and executing motions in unstructured environments that may contain unforeseen obstacles. Meanwhile, many tasks must be performed in a manner that enforces constraints. Humans are aware of such constraints from context and intuition [1]. For example, when pouring water into a bucket with a cup, a person moving the full cup must keep the water approximately level and the water must be poured over the bucket.

Learning from Demonstration (LfD) provides a path to solve this dilemma. LfD has been successfully applied to motion planning and task programming; it allows the human to avoid complex manual programming [2]. Robot control no longer requires building complex models or solving cost functions for planning and optimization; therefore, LfD is friendly for users without professional knowledge. Moreover, by introducing definitions of the operating object and primitive motions, LfD allows the robot's movement to adapt to new environments and reuses previously learned knowledge by relying on motion primitives when composing new tasks [3].

To free the robot from a completely structured working environment, many studies have been conducted in which a robot performs a learned task while avoiding collisions with obstacles [4]. Sampling-based motion planning has been widely implemented in robot collision-free task execution [5], but its high computational cost and instability limit its application. A recent popular idea in LfD is to use probabilistic machine-learning methods in a task-parameterized framework [6], [7]. Although this approach is capable of extracting patterns that are important to a given task and can be generalized to different scenarios, to achieve obstacle avoidance, obstacles must be considered during the demonstration used for model learning [8]. Thus, the availability of training data limits LfD's application because—in all but the simplest domains—the instructor cannot demonstrate all the possible environmental states and their corresponding corrective actions. Consequently, there is a question concerning how the robot should respond when it encounters a new, undemonstrated state (for example, an obstacle category not considered during the demonstration). One approach to dealing with contingencies encountered during execution is to provide additional data by performing interactive corrections [3], [9]. However, this approach increases model complexity and makes dealing flexibly with complex environments and dynamic obstacles difficult [10]. A new approach is therefore needed to enhance the environmental adaptability and obstacle avoidance abilities of probabilistic models. In this paper, we propose a framework that combines probabilistic methods and underlying real-time security controls [11] to reap the benefits of both approaches. This approach allows the probability model to focus on task-related pattern extraction, while the security of task execution is guaranteed by the safety field. The robot can adapt and correct its movement within the task restrictions to cope with dynamic obstacles in real time.

The key contribution of this work is that it enables the robot to avoid collisions in dynamic environments using only a small number of demonstrations yet still being suitable for any kind of obstacle. In contrast, previous studies have the problem of robots performing a task only in the presence of stationary [10], [12], [13] or moving obstacles that are given special consideration during demonstrations [8].

In addition, we extend LfD to allow our approach to perform whole-arm obstacle avoidance in addition to end-effector obstacle avoidance. This aspect of collision avoidance has not received sufficient attention and study in the existing LfD methods. By introducing a new safety assessment, called the Elementary Safety Field (ESF), it is possible to quickly calculate and evaluate the risk of a collision between an operating robot and moving obstacles. Another contribution of this paper is a task- and safety-oriented controller based on a variant of the well-known Closed-Loop Inverse Kinematics (CLIK) algorithm [14]. The control strategy allows redundant manipulators to achieve whole-arm collision-free motion planning in a larger space range beyond the nullspace, while simultaneously performing the task.

The rest of this paper is organized as follows. We review related works in Section II and present the overall method for learning and generalizing tasks in dynamic environments in Section III. Section IV presents the qualitative and quantitative assessments of our method. Finally, a brief summary of the entire work can be found in Section V.

II. RELATED WORKS

Dynamical movement primitives (DMPs) have the ability to model dynamical systems using a set of nonlinear differential equations [15]. A number of methods exist that combine DMPs with a defined control policy for avoiding collisions [10], [16], [17]. For instance, Kardan *et al.* [18] proposed combining the DMPs framework with the virtual impedance method to avoid obstacles in real-time, which enabled a non-oscillatory convergence of the end effector to the desired path. However, traditional DMPs uses only a single demonstration to learn dynamic models. In fact, a more ideal model for task representation can be encoded with multiples demonstrations. To address this problem, a probabilistic representation of movement primitives (ProMPs) [19], [20] was proposed to learn the parameters that describe elementary trajectories, using the concept of phases in the same manner as DMPs. In [19], ProMPs was used to parametrize the desired trajectory distribution of the primitive by a hierarchical Bayesian model with Gaussian distributions. To implement obstacle avoidance, ProMPs need to adapt an existing primitive library by prioritizing additional controllers. Although the approaches based on DMPs-enabled robots to avoid colliding using a predefined security control policy—which is difficult for non-experts to design—it accomplished only space obstacle avoidance of the terminal executor.

Sampling-based motion planners have been widely used by various robotic systems to calculate feasible motion plans in complex environments [21], [22]. Some approaches have combined LfD methods with sample-based motion planning to satisfy learned features in unstructured environments [23], [24]. A prior framework presented by the authors learned the cost-metric-encoded motion features from a set of demonstrations. Then, the sampling-based motion planner computed a collision-free motion plan that globally minimized the cost metric. These methods select an optimal solution from a set of candidates sampled from a feasible configuration space. Although sampling-based methods can satisfy complex path planning scenarios, these methods are computationally inefficient compared to real-time obstacle avoidance methods [25]; therefore, the method was designed for static execution environments [26]. The high computational cost to compute a feasible, obstacle-avoiding trajectory is a major inconvenience for applications in dynamic environments.

In contrast, probabilistic machine-learning methods are widely used for task learning and generalization in LfD. The information provided by only one demonstration is insufficient to reproduce the task: usually, as more training data are used to train the robots, the acquired skills become more robust. Probabilistic models are particularly suitable for analysing multiple demonstrations. To capture the spatial correlations, dynamic time warping (DTW) is commonly used to achieve time alignment of the training data [27]–[29]. The selection of the reference trajectory impacts the results

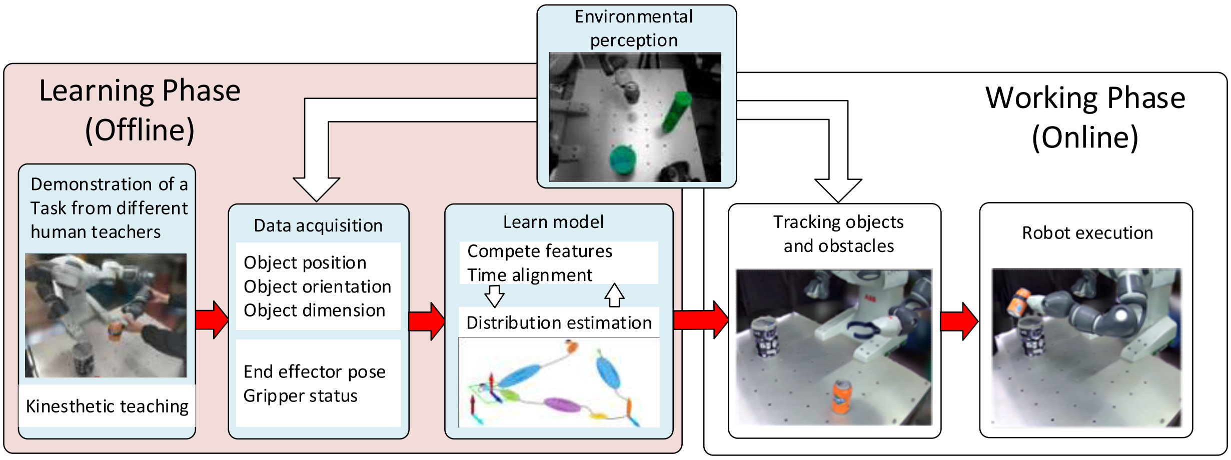


FIGURE 1. Block-diagram of the system components, with task learning shown in orange and execution shown in white. The red arrows indicate execution flow, and the white arrows indicate data flow.

of the DTW application. However, these methods do not consider the influence of the weights of different components (such as cartesian distance and orientation deviation) on the selection of the optimal reference trajectory. This study finds the optimal alignment by defining a similarity metric.

Calinon *et al.* proposed a learning by imitation framework based on a probabilistic model that extracted essential features and generalized human-demonstrated tasks [6], [30]. The demonstrated motion trajectories were modelled by the Gaussian Mixture Model (GMM); then, the generalization trajectory was estimated by Gaussian Mixture Regression (GMR). The GMM/GMR approach can not only cope with changing object reference frames in new situations but also take the obstacles in the environment into account. For instance, Tanwani *et al.* [7] proposed a movement-encoding method based on Hidden Semi-Markov Model (HSMM) that enabled the robot to autonomously replicate a task under different situations, including different starting positions and goal poses and with a simple stationary obstacle. However, to avoid obstacles, these methods must be trained to consider the obstacles during the demonstration stage [9]. In reality, we often do not know what kinds of obstacles will occur in the target environment. To avoid the disadvantages of the GMM/GMR workflow for avoiding obstacles, we take an approach that differs from the previous method.

Khatib presented a real-time obstacle avoidance approach based on the repulsion potential fields concept [11] in which collision avoidance is treated as low-level real-time control. The virtual repulsive force generated between potentially colliding objects and links is converted into the corresponding joint velocities projected to the task nullspace. Motion control based on potential fields is a real-time control method that can effectively guarantee the safe operation of robot [25]. Thus, it is desirable to bridge the gap between the potential fields and probabilistic methods to reap the benefits of both approaches.

Although LfD models and obstacle avoidance have been widely studied, the existing methods do not consider the impact of the kinematic velocity of robot links and moving obstacles. In this paper, we present a safety-oriented and task-oriented control strategy by combining real-time obstacle avoidance and task constraints based on the learned task model. A simple and feasible whole-arm obstacle avoidance algorithm is proposed that greatly reduces the computational overhead. We extract the loose limits of tasks using probabilistic models; this approach provides the basis for a safety control strategy that can avoid arbitrary obstacles in the environment. The proposed approach is more versatile than approaches that integrate task and safety constraints into one task model, and it also reduces the computational difficulty.

III. METHOD OVERVIEW

We assume that the robot can sense the positions of the task-relevant objects, which are called landmarks. During execution, our objective is to compute a collision-free motion plan that applies to the task-relevant objects while satisfying the learned constraints. For the task of pouring water, the robot needs to dynamically adjust the model parameters according to the positions of the cup, bucket and obstacles in the environment, and compute a feasible trajectory. To achieve this goal, the proposed method is divided into two main stages: learning a task model from a set of demonstrations and reactive execution of learned tasks with real-time collision avoidance. Fig. 1 illustrates an overview of the approach. During the learning phase, the operators provide the robot with a series of different demonstrations for the same task to ensure that the obtained model includes sufficient information. There is no need to consider the presence of obstacles for each demonstration. As described in Sec. II-B, first, statistical methods are used to learn a task model; then, the model can be mapped to a new situation, where the task-related objects may appear in different locations.

During the execution phase, the robot first tracks the positions of task-related objects and obstacles in the current environment and calculates the task parameters. We can obtain the safety field of the entire manipulator based on the algorithm proposed in Sec. II-C. The robot then computes a feasible, obstacle-avoiding trajectory based on the control strategy presented in Sec. II-D, causing the entire manipulator to avoid moving obstacles while satisfying the task constraints.

A. DATA ACQUISITION AND PREPROCESSING

To build a robot task model using fewer demonstrations, not only must the demo data be representative, the pre-processing of the demo data is also crucial.

To enable the robot to obtain the required operating skills in a messy environment, we randomly place objects in the environment and record the pose and orientation of the robot's end effector and objects during teaching. Typically, the duration of each demonstration is different. A better robot task model can be obtained through this trajectory pre-processing. DTW is commonly used in the speech recognition field to compare the similarity between two speeches of different lengths. A trajectory is essentially a time series of motion futures; therefore, a trajectory can be processed using the DTW algorithm. The traditional DTW algorithm often requires the reference trajectory to be manually specified, and it assigns the same weight to each dimension. However, each dimension has a different impact on the alignment results at different task stages. This paper proposes a method for automatically obtaining the reference trajectory from multiple presentation trajectories. The similarity between two segments is defined as a "similarity metric" and measured by the total cost of the positional deviation and orientation of the two trajectories after twisting. The deviation between two points νandυ can be calculated as follows:

$$d\left(\nu,\upsilon\right)=\alpha_{\nu}d_{p}\left(\nu,\upsilon\right)+\beta_{\nu}d_{o}\left(\nu,\upsilon\right), \tag{1}$$

where $d_p\left(\nu,\upsilon\right)$ and $d_o\left(\nu,\upsilon\right)$ are, respectively, the cartesian distance and orientation deviation between two track points, and α_ν and β_ν are their respective weights. We generate weights for the position and orientation using the inverse covariance matrix of the generalization trajectory computed by the GMR (which we will introduce in the next section). Here, α_ν^* is equal to the average of the diagonal entry in the inverse covariance matrix corresponding to the current position and β_ν^* corresponds to the orientation. Then, we can obtain

$$\alpha_{\nu}=\frac{\alpha_{\nu}^{*}}{\alpha_{\nu}^{*}+\beta_{\nu}^{*}},\quad \beta_{\nu}=\frac{\beta_{\nu}^{*}}{\alpha_{\nu}^{*}+\beta_{\nu}^{*}}. \tag{2}$$

$DTW(i,j), \forall i,j \in \{1,2,\ldots,M\}$ is used to represent the overall similarity between the two trajectories, where M is the total number of demonstrations. The similarity metric between each demonstration and the rest of the demonstrations is calculated as follows:

$$similarity(i)=\sum_{j=1}^{M}DTW(i,j), \forall i\in\{1,2,\ldots,M\}. \tag{3}$$

The demonstration with the minimum similarity is selected as the reference trajectory. Therefore, we traverse the trajectories for DTW alignment and obtain a GMR as the reference trajectory. Then, we calculate the similarity metric of all the trajectories using Formula (3).

Next, we will use the behaviour characterization based on the Gaussian Mixture Model to code the teaching data, which is mainly composed of two parts: learning the model parameters and generalizing to new scenarios.

B. MODEL CONSTRUCTION

The demo data set can be represented as $\{\boldsymbol{X}^j \in \mathbb{R}^{D\times N}|j = 1,\ldots,M\}$, which corresponds to matrices composed of D-dimensions of the observations at N time steps. In this paper, $D = 8$, including the time index, cartesian position and attitude quaternion.

To construct task models related to environmental objects, the cartesian coordinate and the orientation (in quaternions) of the gripper's tip at time step n in the world frame is expressed as $\boldsymbol{a}_n$. In the object coordinate system p, it can be expressed as $\boldsymbol{a}_n^p$. Constructing the equation $\boldsymbol{a}_n = \boldsymbol{A}_{n,p}\boldsymbol{a}_n^p + \boldsymbol{b}_{n,p}$ makes it easier to calculate the Gaussian distribution products, where $\boldsymbol{b}_{n,p}$ represents the offset of the object coordinate from the base coordinate, and $\boldsymbol{A}_{n,p}$ is the transformation matrix. The parameters of the model with K components are defined by $\{\pi_i, \{\boldsymbol{\mu}_i^j, \boldsymbol{\Sigma}_i^j\}_{j=1}^P | i = 1,\ldots,K\}$, where π_i represent the mixing coefficients and $\boldsymbol{\mu}_i^j$, $\boldsymbol{\Sigma}_i^j$ are, respectively, the centre and covariance matrix of the *i-th* Gaussian component at frame j. The parameters are determined by expectation-maximization (EM) continuous iterative learning until the model finally converges. To prevent the algorithm from becoming trapped in a local optimum, k-means clustering is used for parameter initialization.

The expected multi-constraint probability distribution can be represented by the product of each constraint probability distribution. After obtaining the GMM model of the task in each object coordinate system, the robot can acquire the task model according to the objects' positions and postures in the new environment as follows:

$$\begin{aligned}
\mathcal{N}(\boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i}) &\propto \prod_{j=1}^{P}\mathcal{N}(\boldsymbol{A}_{n,j}\boldsymbol{\mu}_i^j + \boldsymbol{b}_{n,j}, \boldsymbol{A}_{n,j}\boldsymbol{\Sigma}_i^j\boldsymbol{A}_{n,j}^{\mathrm{T}})\\
\boldsymbol{\Sigma}_{n,i} &= \left(\sum_{j=1}^{P}\left(\boldsymbol{A}_{n,j}\boldsymbol{\Sigma}_i^j\boldsymbol{A}_{n,j}^{\mathrm{T}}\right)^{-1}\right)^{-1}\\
\boldsymbol{\mu}_{n,i} &= \boldsymbol{\Sigma}_{n,i}\sum_{j=1}^{P}\left(\boldsymbol{A}_{n,j}\boldsymbol{\Sigma}_i^j\boldsymbol{A}_{n,j}^{\mathrm{T}}\right)^{-1}\left(\boldsymbol{A}_{n,j}\boldsymbol{\mu}_i^j + \boldsymbol{b}_{n,j}\right).
\end{aligned} \tag{4}$$

Using the GMM parameters computed by (2), a position and orientation reference is retrieved by the GMR at each time step. The GMM can be expressed as the joint distribution $\mathcal{P}(\boldsymbol{\xi}_n^I, \boldsymbol{\xi}_n^O)$, where $\boldsymbol{\xi}_n^I$ is the time step and $\boldsymbol{\xi}_n^O$ is the position and orientation. The conditional probability $\mathcal{P}(\boldsymbol{\xi}_n^O|\boldsymbol{\xi}_n^I)$ is then estimated as an output distribution $\mathcal{N}(\hat{\boldsymbol{\xi}}_n^O, \hat{\boldsymbol{\Sigma}}_n^O)$ that is also Gaussian [6].

C. OBSTACLE AVOIDANCE ALGORITHM

In human-robot cooperation, we must not only consider the movement of the end effector but also the potential collision links. The influence of obstacles on motion planning is reflected not only by position but also by shape, size and velocity. Due to factors such as computing power and perceptual equipment, this fact is often overlooked. We proceed from the simplest definition, gradually expand to the rigid body, and finally, achieve real-time motion obstacle avoidance of the robot arm at a reduced computational cost.

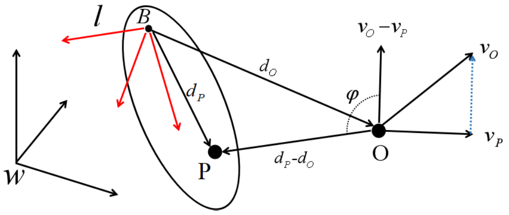


FIGURE 2. Schematic diagram of elementary safety field calculation. Point "O" represents the moving rigid body, namely, the "source of danger", expressed in the reference frame *l*, drawn in red. Point "P" represents any point on the robot link, "B" is the origin of the robot joint frame *l*, and *w* is the world frame.

1) BASIC DEFINITION

We regard the obstacle as a point O and the link of the robot arm as a rigid body in the world frame w (see Fig. 2). A point P on the robot link is represented as $\boldsymbol{d_P} = (x_P\, y_P\, z_P)^T$ and $\boldsymbol{v_P}$ in the local reference frame l, which is the robot joint coordinate system, and the joint rotation axis is its z-axis. According to the robot kinematics, the velocity of the origin of the l coordinate system is calculated as $\boldsymbol{v_B^w}$, and T is the transformation matrix between l and w. Then, the velocity of points B and P in the l coordinate system are, respectively,

$$\boldsymbol{v_B^l} = {}_l^w \boldsymbol{T}^{-1} * \boldsymbol{v_B^w}, \quad \boldsymbol{v_P} = \boldsymbol{v_B^l} + \boldsymbol{\omega} \times \boldsymbol{d_P}. \tag{5}$$

An obstacle is expressed as $\boldsymbol{d_O} = (x_O\, y_O\, z_O)^T$ and $\boldsymbol{v_O}$ in the local frame l. The relative velocity of two points is $\boldsymbol{v} = \boldsymbol{v_O} - \boldsymbol{v_P}$. The angle between the vectors $\boldsymbol{d_P} - \boldsymbol{d_O}$ and $\boldsymbol{v}$ is defined as $\varphi = \angle\left(\boldsymbol{d_P} - \boldsymbol{d_O}, \boldsymbol{v}\right) \in [-\pi, \pi)$. The φ can be computed as follows:

$$\cos\varphi = \frac{\langle \boldsymbol{d_P} - \boldsymbol{d_O}, \boldsymbol{v} \rangle}{\|\boldsymbol{d_P} - \boldsymbol{d_O}\| \, \|\boldsymbol{v}\|}. \tag{6}$$

We define an Elementary Safety Field (ESF) by the point on the manipulator and the moving obstacle in the environment. ESF is represented by the following scalar function:

$$ESF\left(\boldsymbol{d_P}, \boldsymbol{d_O}, \boldsymbol{v}\right) = \|\boldsymbol{d_P} - \boldsymbol{d_O}\| - \alpha\cos\varphi\,\|\boldsymbol{v}\|, \tag{7}$$

where α is a normal number positively related to the control cycle. When $ESF\left(\boldsymbol{d_P}, \boldsymbol{d_O}, \boldsymbol{v}\right) < 0$, a collision is imminent, and the robot movement should be stopped.

2) EXTENSION TO RIGID BODIES

To achieve obstacle avoidance of the entire arm, it is necessary to calculate the minimum ESF of each link and accumulate the obstacle-avoiding joint angular velocity generated by the minimum ESF of each link. The links of most series manipulators such as the UR and YUMI robots can be favourably considered as cylinders. Therefore, we can approximate the robot links as cylinders and pay attention only to the points on the cylinder axis. Further, order to calculate the velocities of the points on the cylindrical axis more efficiently, we divide the links of the robot arm into two types: the Z axis of the l coordinate system along the cylinder's axis and the Y axis of the l coordinate system along the axis direction.

In condition 1, joint angular velocity does not affect the velocity of point P, which means that $\boldsymbol{v_P} = \boldsymbol{v_B^l}$. The ESF at the point closest to the obstacle is minimal. The nearest point S and the shortest distance $\|\boldsymbol{d_P} - \boldsymbol{d_O}\|_S$ can be obtained according to the distance equation from a dot to a straight line; therefore, we can obtain the minimum ESF as follows:

$$ESF_S = \|\boldsymbol{d_P} - \boldsymbol{d_O}\|_S - r - \alpha\cos\varphi \left\| \boldsymbol{v_B^l} - \boldsymbol{v_O} \right\|, \tag{8}$$

where r is the diameter of the enveloping cylinder.

In condition 2, the coordinates of the point on the axis of the cylinder can be expressed as $(0, y_P, 0)$, where $\boldsymbol{v_P}$ is affected by the joint angular velocity and can be expressed as

$$\begin{aligned} \boldsymbol{v_P} &= \left(v_{B,x}^l, v_{B,y}^l, v_{B,z}^l\right)^{\boldsymbol{T}} + (-\omega * y_P, 0, 0)^{\boldsymbol{T}} \\ &= \left(v_{B,x}^l - \omega * y_P, v_{B,y}^l, v_{B,z}^l\right)^{\boldsymbol{T}} \end{aligned} \tag{9}$$

$$\|\boldsymbol{d_P} - \boldsymbol{d_O}\| = \sqrt{x_O^2 + (y_O - y_P)^2 + z_O^2}. \tag{10}$$

By deriving from Formula (5), we obtain

$$\begin{aligned} &ESF \\ &= \sqrt{x_O^2 + (y_O - y_P)^2 + z_O^2} - r - \alpha cos\varphi \\ &\times \sqrt{\left(v_{B,x}^l - \omega * y_P - v_{O,x}^l\right)^2 + (v_{B,y}^l - v_{O,y}^l)^2 + \left(v_{B,z}^l - v_{O,z}^l\right)^2}. \end{aligned} \tag{11}$$

Then, by computing the partial derivative of ESF with respect to y_p, we let $\frac{\partial ESF}{\partial y_P} = 0$ and obtain $y_p = y_1, y_2, \ldots y_n \in (0, Y)$, where Y is the length of the link. Finally, we can find ESF_S.

We want the avoidance response of the robot arm to be a function of the ESF_S such that as the arm moves closer to an obstacle the response becomes stronger:

$$\boldsymbol{\Delta p_S} = \begin{cases} \eta\left(\rho_0 - ESF_S\right)^2 \dfrac{\partial \boldsymbol{ESF_S}}{\partial \boldsymbol{x}}, & ESF_S \le \rho_0 \\ \boldsymbol{0}, & ESF_S > \rho_0, \end{cases} \tag{12}$$

where ρ_0 is the threshold distance at which the avoidance function is activated, $\frac{\partial \boldsymbol{ESF_S}}{\partial \boldsymbol{x}}$ is the partial derivative of the $\boldsymbol{ESF_S}$ from the point on the arm to the obstacle, and η is a gain term. For a single desired displacement $\boldsymbol{\Delta p_S}$ of the point S located on the manipulator's arm, the corresponding increment of the joint angle $\boldsymbol{\Delta q_S}$ is obtained by

$$\boldsymbol{\Delta q_S} = k_p \boldsymbol{J}^{\boldsymbol{T}}_{\boldsymbol{S,v}}(\boldsymbol{q})\boldsymbol{\Delta p_S}, \tag{13}$$

where $\boldsymbol{J}^{\boldsymbol{T}}_{\boldsymbol{S,v}}(\boldsymbol{q})$ represents the first three rows of the Jacobian matrix $\boldsymbol{J}^{\boldsymbol{T}}_{\boldsymbol{S}}(\boldsymbol{q})$ associated with point S, and k_p is a positive real parameter. By dividing the sampling time on both sides of Formula (11), the joint velocity command can be expressed as $\dot{\boldsymbol{q}}_{\boldsymbol{S}} = k_v\mathbf{J}^{\boldsymbol{T}}_{\boldsymbol{S,v}}(\boldsymbol{q})\boldsymbol{\Delta p_S},$. The robot's overall velocity command accumulates the evasive velocity caused by all the points S on the robot arm:

$$\dot{\boldsymbol{q}}_{\boldsymbol{O}} = \sum_{S} k_v\mathbf{J}^{\boldsymbol{T}}_{\boldsymbol{S,v}}(\boldsymbol{q})\boldsymbol{\Delta p_S}. \tag{14}$$

D. MISSION-ORIENTED SECURITY CONTROL STRATEGY

In Sec. II-B, we obtained the desired position and orientation of the end effector $\boldsymbol{x_n}$ at each sample time step n. The CLIK algorithm can be used to combine a task-related command with the obstacle avoidance command. The standard CLIK algorithm is shown in Fig. 3.

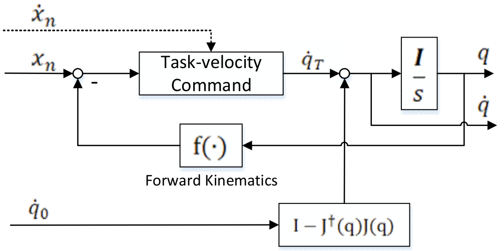


FIGURE 3. A diagram of the standard CLIK algorithm, using the signal $\dot{q}_O$ computed by *ESF* to avoid collisions.

The task-related velocity can be calculated by $\dot{\boldsymbol{q}}_{\boldsymbol{T}} = \boldsymbol{J}^{\dagger}(\boldsymbol{q})\boldsymbol{K_e}(\boldsymbol{x_n} - \boldsymbol{x})$, where $\boldsymbol{K_e}$ is a positive definite symmetric matrix. However, in reality, algorithms often fail to meet the demand for obstacle avoidance. We added the controllable variable k to the standard CLIK algorithm, which is related to the smallest ESF_S of all links.

$$\dot{\boldsymbol{q}} = k\dot{\boldsymbol{q}}_{\boldsymbol{T}} + [\boldsymbol{I} - k\boldsymbol{J}^{\dagger}(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})]\dot{\boldsymbol{q}}_{\boldsymbol{O}}. \tag{15}$$

The obstacle avoidance velocity command causes the position of the end effector to deviate from the desired position. Since $\boldsymbol{x_n}$ is the optimal estimate of the distribution $\mathcal{N}(\hat{\boldsymbol{\xi}}^{\boldsymbol{O}}_{\boldsymbol{n}}, \hat{\boldsymbol{\Sigma}}^{\boldsymbol{O}}_{\boldsymbol{n}})$, $\boldsymbol{x_n}$ should be within a certain range. Let $\{x^j_n | j = 1, \ldots, D\}$ represent a variable for each dimension of $\boldsymbol{x_n}$, and σ^j_n represent the standard deviation of the dimension j obtained from the covariance matrix $\hat{\boldsymbol{\Sigma}}^{\boldsymbol{O}}_{\boldsymbol{n}}$. Then, the value range can be defined as follows:

$$x^j_n - \lambda\sigma^j_n < x^j < x^j_n + \lambda\sigma^j_n, \tag{16}$$

with a positive constant λ.

The k value determines the proportion of task-related velocity and evasive velocity; thus, it affects the obstacle avoidance and task execution performance. Defining the thresholds ρ_0, ρ_1 and $ESF_{min} = \min(ESF^l_S | l = 1, \ldots, L)$, when ESF_{min} is less than ρ_0, the standard CLIK algorithm is activated, and when ESF_{min} is less than ρ_1, the variant CLIK algorithm (15) is activated. The continuous decrease of ESF_{min} indicates that the control strategy can no longer avoid the obstacle. Therefore, the value of k should be changed to increase the weight of the obstacle avoidance velocity command. We iteratively update the k values at the current time based on the k_{last} and ESF^{last}_{min} values from the last update cycle and the current $ESF^{current}_{min}$, as shown below:

$$k = k_{last} + k_{last}\frac{ESF^{current}_{min} - ESF^{last}_{min}}{ESF^{last}_{min}} = k_{last}\frac{ESF^{current}_{min}}{ESF^{last}_{min}}. \tag{17}$$

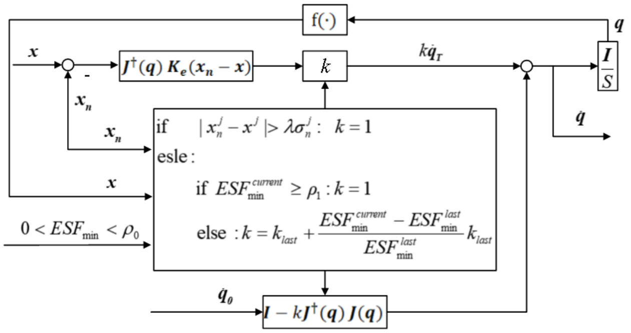

f(·)
q
x
-
$J^{\dagger}(q)\,K_e(x_n - x)$
k
$k\dot{q}_T$
$\frac{I}{S}$
x_n
x_n
$\dot{q}$
if $|x^j_n - x^j| > \lambda\sigma^j_n$: $k = 1$
esle :
x
if $ESF^{current}_{min} \geq \rho_1 : k = 1$
$0 < ESF_{min} < \rho_0$
else $: k = k_{last} + \frac{ESF^{current}_{min} - ESF^{last}_{min}}{ESF^{last}_{min}} k_{last}$
$\dot{q}_0$
$I - kJ^{\dagger}(q)\,J(q)$

FIGURE 4. A diagram of the variant CLIK algorithm. In each control cycle, the robot calculates the task-related velocity $\dot{q}_T$ and evasive velocity $\dot{q}_O$. Then, Formula (15) is used to calculate the final control command. To select the *k* value, we need to consider both the task constraints and the obstacle avoidance requirement. If the end-effector is beyond the expected range, we set *k* = 1 to force it back into the desired trajectory. When *ESF*$_{min}$ is less than ρ_1, which implies that the robot is at risk of collision, we should update *k* according to the variation trend of *ESF*$_{min}$ and the last *k* value *k*$_{last}$.

This process causes the robot to dynamically adjust the k value based on its previous state and the distance change trend and guarantees that k remains in the range of 0 to 1. Fig. 4 shows a diagram of the framework, while Table 1 summarizes our approach and its different stages.

IV. EXPERIMENTS

A YUMI 14-DOF dual-arm cooperative robot is used in this experiment. The goal is for the robot to grasp a can and track a bucket to imitate the action of people pouring water while avoiding obstacles in the operation space. We track the positions of target objects and obstacles in real time through vision sensor, as the robot performs tasks in a dynamic environment. We provide 12 successful presentations with different can and bucket positions. The object manipulation task consists of the following actions: picking up the can with the left robot arm, keeping the can vertical and tracking the bucket position, and then pouring the contents of the can into the bucket, as shown in Figure 1.

In this experiment, three candidate frames ($P = 3$) are considered, namely, the frames representing the locations of

TABLE 1. Overall framework of complete system.

1.Task demonstrations

- Get demo data $\left\{ \boldsymbol{X}^j \in \mathbb{R}^{D\times N} | j = 1,\ldots,M \right\}$

-Collect $\boldsymbol{A}_{n,p}, \boldsymbol{b}_{n,p}$ of different object frames p

-Perform data pre-processing using DTW algorithm

-Use the similarity metric to find the optimal DTW alignment

2.Model construction

-Determine K by Bayesian Information Criteria

-Optimize $\left\{ \pi_i, \left\{ \boldsymbol{\mu}_i^j, \boldsymbol{\Sigma}_i^j \right\}_{j=1}^{P} | i = 1,\ldots,K \right\}$ by EM

3.Reproduction

For $n \to 1$ to N (for each time step)

-Collect/select ξ_n^I and $\{\boldsymbol{A}_{n,p}, \boldsymbol{b}_{n,p}\}_{p=1}^{P}$

-Use (4) to estimate the temporary GMM $\mathcal{N}\left(\boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i}\right)$, and retrieve $\hat{\xi}_n^O$ by the GMR, $\mathcal{N}\left(\hat{\xi}_n^O, \hat{\boldsymbol{\Sigma}}_n^O\right)$

-Compute the current $ESF_{min}^{current}$ in every update cycle (below 30ms) and use (17) to compute the k value

-use (15) combining the task and avoidance velocity

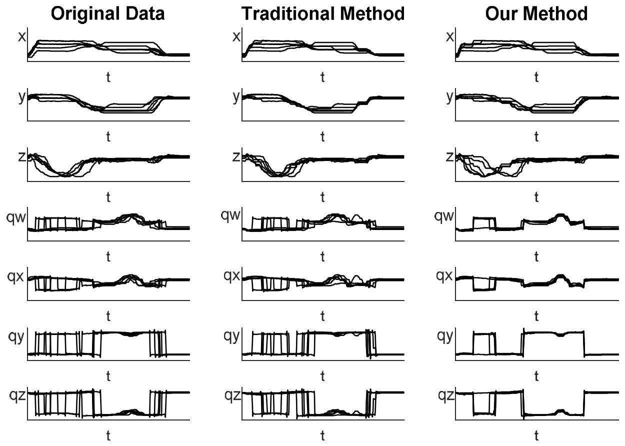


FIGURE 5. The teaching trajectories and the trajectories alignment results using the traditional DTW method and method proposed in this paper respectively. The traditional method assigns the same weight to position deviation and orientation deviation, while our method dynamically changes the weights according to the importance of the components shown in task models. In the third column, we can see that the trajectory gets better alignment in the quaternion components.

the main objects (can, bucket) and the robot's basal coordinates (world frame). During task reproduction, the objects' positions are obtained in real time by a Primese 1.09 camera and transferred to the model. We can obtain the temporary GMM parameters using (4) and a new attractor location is retrieved in each time step n. As mentioned in Sec. II-A, in previous works, DTW is unable to obtain the optimal reference trajectory, and it assigns the same weight to each dimension. Fig. 5 shows the differences in the results when using the traditional DTW and using our method to process data. As shown, the trajectories achieve relatively good alignment in the x, y and z dimensions but are unsuitable in the orientation dimensions when using the traditional method. However, the data alignment results are more consistent in the orientation dimensions when using the proposed method, which reflects the consistent movement characteristics (keeping the can upright and pouring water).

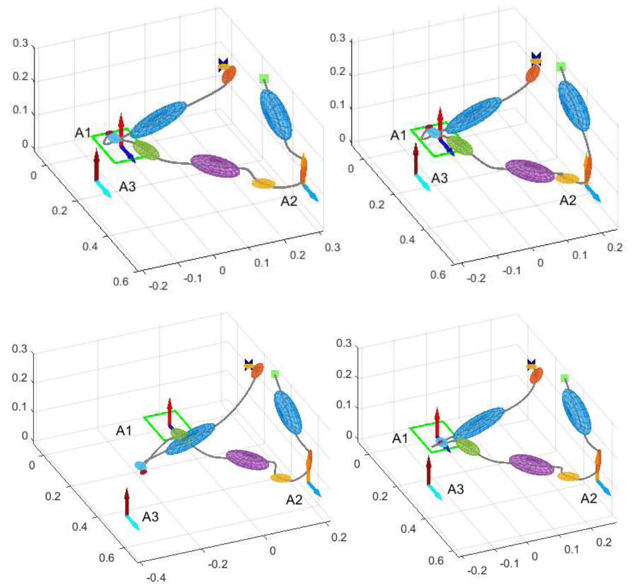


FIGURE 6. YUMI robot performs picking and pouring task with different object frames. After probabilistic model with 10 Gaussian components has been learned from training data, the robot acquire the new GMM task model according the objects' position and posture in the new environment. A1 is the robot basal coordinates and A2 and A3 represent can and bucket coordinate systems, respectively.

Fig. 6 shows the generalization results of the attractor trajectory computed from (4) with different object frames. The smooth movements are reproduced by learned model and the task constraints are extrapolated to new situations. Note that the reproductions are locally consistent when the robot approaches the initial location of the can (frame A2); the end effector dumps the can's content after it reaches the bucket position (frame A3). This is reflected by the small and narrow ellipsoids at the corresponding stages of the task. The attractor trajectories with large and broad ellipsoids at the other stages are not consistent and contain no useful task information; therefore, we can use these loose restrictions to achieve obstacle avoidance. We apply the GMR to obtain a position and orientation reference at each time step, as shown in Fig. 7.

After learning, two types of tests were conducted to test the method's reproduction and generalization performance. We test the generalization capabilities of the method by placing the can and bucket at different locations on the desktop. In addition, we quickly move the bucket to a different random location when the manipulator approaches it (Fig. 8(a), (b) and (c)). A test case is considered

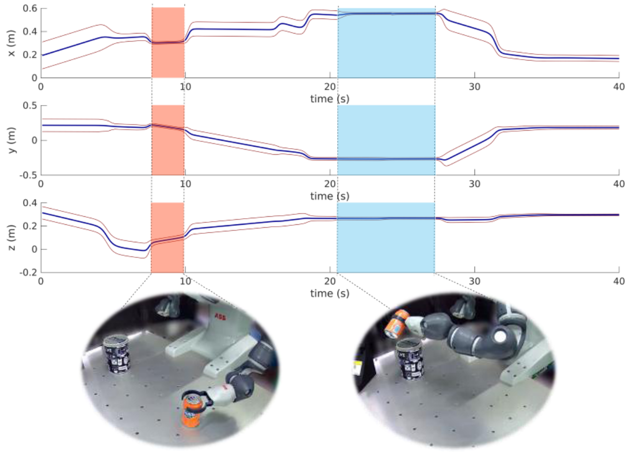


FIGURE 7. The generalized trajectory computed by the GMR (the blue lines). The value range computed by (16) when $\lambda = 1$ is represented by the red lines. We can see that the trajectories are highly constrained by time between 8 s and 10 s (when reaching for the can). The trajectories are also highly constrained when pouring water into the bucket. The snapshots below the graphs illustrate that the robot performs a reproduction attempt according to the extracted constraints.

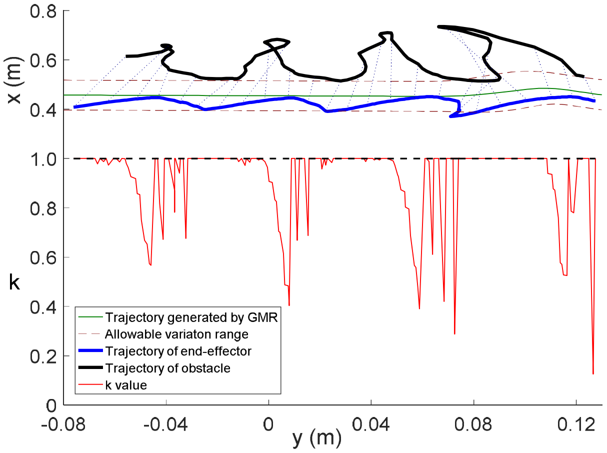


FIGURE 9. The upper half of the Fig. shows the corresponding trajectory of the robot's end-effector (blue line) and the moving obstacle (black line) in the x, y plane of the robot basal coordinates. As the obstacle approaches the robot arm, the robot performs an obstacle avoidance response within the allowable range (denoted by the dotted lines). The red line in the bottom half of the Fig. shows how the *k* value varies with the position of the end effector.

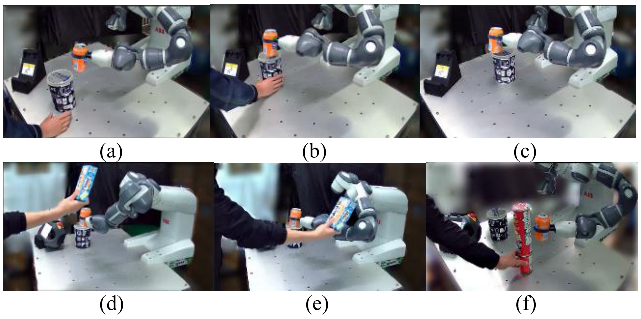


FIGURE 8. a), (b) and (c) show the snapshots that robot tracks the moving bucket while keeping the can straight up. (d) and (e) show that the robot performs nullspace motion to avoid collision using the ESF when the obstacle (blue box) moves to the robot link. (f) is the snapshot that the robot avoids collision between the can and the moving obstacle (red cylinder) which is moved intentionally to cause a collision.

successful if the robot: 1) grasps the can and transfers it to the bucket without spilling and 2) accurately tracks and aligns the bucket. The probability of success on these tests reached eighty percent. In the failed tests, the robot was unable to grasp the can because the deviation of the attraction point from the target exceeded the threshold.

The second test consisted of avoiding moving obstacles to evaluate the robot's obstacle avoidance capability. In this experiment, we set the parameters to $\rho_0 = 0.3$, $\rho_1 = 0.2$ and $\lambda = 1$. First, we verify the effectiveness of the obstacle avoidance algorithm proposed in Sec. II-C. Fig. 8(d) and (e) shows two snapshots of the robot performing the nullspace motion to prevent possible collisions with a moving obstacle. Next, the robot must conduce motion planning in the dynamic environment, where a cylindrical red object is being moved intentionally to cause a collision with the manipulator (see Fig. 8(f)). This experiment verifies the effectiveness of our task-oriented security control strategy with moving obstacles. Fig. 9 shows the corresponding trajectory of the end effector and the obstacle and illustrates that when the obstacle approaches the robot arm, the robot performs an obstacle avoidance response within the allowable range. We also show how the *k* value varies with the end effector position.

We can conclude that although the *k* value varies drastically due to the limitations of all dimensions, obstacle avoidance is still accomplished. In addition, all the computation was performed on a PC with an eight-core Intel Core i7-7700 processor, 8 GB of RAM and a real-time Linux-based operating system. The minimum and maximum time durations required to compute the proposed control strategy were 13.85 [ms] and 28.34 [ms].

V. CONCLUSIONS

In this paper, we propose a safety- and task-oriented control strategy for redundant manipulators based on a probabilistic model and underlying real-time security controls. Unlike previous methods, this approach uses the concept of an elementary safety field to enable the entire robot arm robot to achieve obstacle avoidance and respond quickly to moving obstacles. We propose a modified DTW algorithm that uses a similarity metric defined in this paper to find the optimal reference trajectory and alignment results. The ESF is a novel concept in the robot safety field that can be used to predict collisions between the entire manipulator and moving obstacles. In this study, we use the ESF to calculate the desired displacement of the manipulator to avoid obstacles. Furthermore, based on the learned task-parametrized probabilistic model, kinematic redundancy and the space with no task restrictions are exploited to simultaneously improve task performance and avoid moving obstacles.

The presented experimental results show that the approach proposed in this paper not only successfully models the task but also enables the robot to avoid a moving obstacle while still satisfying the task constraints. The proposed framework is suitable for all types of obstacles and has good real-time performance. Moreover, compared with the traditional CLIK algorithm, the control framework proposed in this paper achieves a wider range of obstacle avoidance. This aspect is crucial for human-robot coexistence when human and robot operating spaces overlap.

VI. ACKNOWLEDGMENT

(Gangfeng Liu and Caiwei Song are co-first authors.)

REFERENCES

[1] G. Maeda, M. Ewerton, G. Neumann, R. Lioutikov, and J. Peters, ''Phase estimation for fast action recognition and trajectory generation in human–robot collaboration,'' *Int. J. Robot. Res.*, vol. 36, nos. 13–14, pp. 1579–1594, Dec. 2017, doi: 10.1177/0278364917693927.

[2] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, ''Imitation learning: A survey of learning methods,'' *ACM Comput. Surv.*, vol. 50, no. 2, 2017, Art. no. 21.

[3] S. Niekum, S. Osentoski, G. Konidaris, S. Chitta, B. Marthi, and A. G. Barto, ''Learning grounded finite-state representations from unstructured demonstrations,'' *Int. J. Robot. Res.*, vol. 34, no. 2, pp. 131–157, Feb. 2015, doi: 10.1177/0278364914554471.

[4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, ''A survey of robot learning from demonstration,'' *Robot. Auto. Syst.*, vol. 57, no. 5, pp. 469–483, 2009, doi: 10.1016/j.robot.2008.10.024.

[5] A. Short, Z. Pan, N. Larkin, and S. van Duin, ''Recent progress on sampling based dynamic motion planning algorithms,'' in *Proc. IEEE AIM*, Jul. 2016, pp. 1305–1311.

[6] S. Calinon, D. Bruno, and D. G. Caldwell, ''A task-parameterized probabilistic model with minimal intervention control,'' in *Proc. ICRA*, May/Jun. 2014, pp. 3339–3344.

[7] A. K. Tanwani and S. Calinon, ''Learning robot manipulation tasks with task-parameterized semitied hidden semi-Markov model,'' *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 235–242, Jan. 2016, doi: 10.1109/LRA.2016.2517825.

[8] M. G. E. Amir and M. Ragaglia, ''Robot learning from demonstrations: Emulation learning in environments with moving obstacles,'' *Robot. Auton. Syst.*, vol. 101, pp. 45–56, Mar. 2018, doi: 10.1016/j.robot.2017.12.001.

[9] M. Karlsson, A. Robertsson, and R. Johansson, ''Autonomous interpretation of demonstrations for modification of dynamical movement primitives,'' in *Proc. ICRA*, May/Jun. 2017, pp. 316–321.

[10] A. M. Ghalamzan, C. Paxton, G. D. Hager, and L. Bascetta, ''An incremental approach to learning generalizable robot tasks from human demonstration,'' in *Proc. ICRA*, May 2015, pp. 5616–5621.

[11] O. Khatib, ''Real-time obstacle avoidance for manipulators and mobile robots,'' *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, 1986, doi: 10.1177/027836498600500106.

[12] A. Rai, G. Sutanto, F. Meier, and S. Schaal, ''Learning feedback terms for reactive planning and control,'' in *Proc. ICRA*, May/Jul. 2017, pp. 2184–2191.

[13] N. Ratliff, M. Toussaint, and S. Schaal, ''Understanding the geometry of workspace obstacles in motion optimization,'' in *Proc. ICRA*, May 2015, pp. 4202–4209.

[14] B. Siciliano, ''A closed-loop inverse kinematic scheme for on-line joint-based robot control,'' *Robotica*, vol. 8, no. 3, pp. 231–243, Jul. 1990, doi: 10.1017/S0263574700000096.

[15] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, ''Dynamical movement primitives: Learning attractor models for motor behaviors,'' *Neural Comput.*, vol. 25, no. 2, pp. 328–373, 2013, doi: 10.1162/NECOa00393.

[16] 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 *Proc. ICRA*, May 2009, pp. 2587–2592.

[17] P. Kormushev, S. Calinon, and D. G. Caldwell, ''Robot motor skill coordination with EM-based reinforcement learning,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 3232–3237.

[18] I. Kardan, A. Akbarzadeh, and A. M. Mohammadi, ''Real-time velocity scaling and obstacle avoidance for industrial robots using fuzzy dynamic movement primitives and virtual impedances,'' *Ind. Robot*, vol. 45, no. 1, pp. 110–126, 2018, doi: 10.1108/IR-02-2017-0035.

[19] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, ''Using probabilistic movement primitives in robotics,'' *Auto. Robots*, vol. 42, no. 3, pp. 529–551, Mar. 2018, doi: 10.1007/s10514-017-9648-7.

[20] A. Paraschos, R. Lioutikov, J. Peters, and G. Neumann, ''Probabilistic prioritization of movement primitives,'' *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2294–2301, Oct. 2017, doi: 10.1109/LRA.2017.2725440.

[21] S. Karaman and E. Frazzoli, ''Sampling-based algorithms for optimal motion planning,'' *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011, doi: 10.1177/0278364911406761.

[22] M. Zucker, N. Ratliff, and A. D. Dragan, ''CHOMP: Covariant Hamiltonian optimization for motion planning,'' *Int. J. Robot. Res.*, vol. 32, nos. 9–10, pp. 1164–1193, Aug. 2013, doi: 10.1177/0278364913488805.

[23] C. Bowen, G. Ye, and R. Alterovitz, ''Asymptotically optimal motion planning for learned tasks using time-dependent cost maps,'' *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 171–182, Jan. 2015, doi: 10.1109/TASE.2014.2342718.

[24] C. Bowen and R. Alterovitz, ''Closed-loop global motion planning for reactive execution of learned tasks,'' in *Proc. IROS*, Sep. 2014, pp. 1754–1760.

[25] B. Lacevic, P. Rocco, and A. M. Zanchettin, ''Safety assessment and control of robotic manipulators using danger field,'' *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1257–1270, Oct. 2013, doi: 10.1109/TRO.2013.2271097.

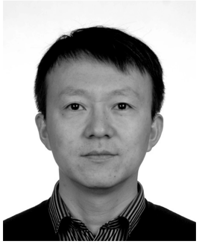
[26] T. Osa, A. M. G. Esfahani, R. Stolkin, R. Lioutikov, J. Peters, and G. Neumann, ''Guiding trajectory optimization by demonstrated distributions,'' *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 819–826, Apr. 2017, doi: 10.1109/LRA.2017.2653850.

[27] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, ''Statistical dynamical systems for skills acquisition in humanoids,'' in *Proc. 12th IEEE-RAS Int. Conf. Humanoid Robots*, Dec. 2012, pp. 323–329.

[28] J. Jiang, X. Yuan, W. Shuxin, and L. Ke, ''Evaluation of robotic surgery skills using dynamic time warping,'' *Comput. Methods Programs Biomed.*, vol. 152, pp. 71–83, Dec. 2017, doi: 10.1016/j.cmpb.2017.09.007.

[29] C. Bowen and R. Alterovitz, ''Asymptotically optimal motion planning for tasks using learned virtual landmarks,'' *IEEE Robot. Autom. Lett.*, vol. 1, no. 2, pp. 1036–1043, Jul. 2016, doi: 10.1109/LRA.2016.2530877.

[30] T. Alizadeh, S. Calinon, and D. G. Caldwell, ''Learning from demonstrations with partially observable task parameters,'' in *Proc. ICRA*, May/Jun. 2014, pp. 3309–3314.



GANGFENG LIU (M'11) was born in Laixi, Qingdao, Shandong, China, in 1980. He received the B.S., M.S., and Ph.D. degrees in mechanical engineering from the Harbin Institute of Technology, Harbin, Heilongjiang, China, in 2003, 2005, and 2010, respectively.

From 2010 to 2016, he was a Research Assistant with the Robot Institute. Since 2016, he has been an Assistant Professor with the Mechanical Engineering Department, Harbin Institute of Technology. He has authored over 40 articles and over 30 inventions. His research interests include autonomous learning, compliant control, and mechanism optimization.

CAIWEI SONG was born in Jimo, Qingdao, Shandong, China, in 1992. He received the B.S. degree in mechanical engineering from Shandong University, Jinan, China, in 2015.

He is currently pursuing the Ph.D. degree at the State Key Laboratory of Robotics and System, Harbin Institute of Technology. His research interests include robotic automated planning, robotic control, and cooperative manipulation.

XIZHE ZANG (M'09) received the M.S. degree in mechatronics engineering and the Ph.D. degree in mechanical engineering from the Harbin Institute of Technology, China, in 1999 and 2005, respectively. He is currently an Associate Professor with the School of Mechatronics Engineering, Harbin Institute of Technology. His research interests include bionic robots, multi-robot system, and teleoperation.

JIE ZHAO (M'05) received the B.S. and Ph.D. degrees in mechatronics engineering from the Harbin Institute of Technology (HIT), Harbin, China, in 1990 and 1996, respectively.

He is currently a Professor with the School of Mechatronics Engineering, HIT, where he is also the Director of the State Key Laboratory of Robotics and System.

His research interests include industrial robots and bionic robots. He is the Leader of the Subject Matter Expert Group of Intelligent Robot in National 863 Program supervised by the Ministry of Science and Technology of China.