TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 02/10 pp654–662
DOI: 10.26599/TST.2018.9010096
Volume 24, Number 6, December 2019

Skill Learning for Human-Robot Interaction Using Wearable Device

Bin Fang*, Xiang Wei, Fuchun Sun, Haiming Huang, Yuanlong Yu, and Huaping Liu

Abstract: With the accelerated aging of the global population and escalating labor costs, more service robots are needed to help people perform complex tasks. As such, human-robot interaction is a particularly important research topic. To effectively transfer human behavior skills to a robot, in this study, we conveyed skill-learning functions via our proposed wearable device. The robotic teleoperation system utilizes interactive demonstration via the wearable device by directly controlling the speed of the motors. We present a rotation-invariant dynamical-movement-primitive method for learning interaction skills. We also conducted robotic teleoperation demonstrations and designed imitation learning experiments. The experimental human-robot interaction results confirm the effectiveness of the proposed method.

Key words: skill learning; interaction; teleoperation; dynamical movement primitive

1 Introduction

In recent years, with the accelerated aging of the global population and escalating labor costs, the demand for service robots has also increased. These robots can help elderly and disabled people to perform daily tasks, such as wiping a table, massaging a limb, organizing household items, and delivering items, to name a few. To perform these various daily-life tasks, the robot must master a range of operating skills and intelligently combine skill units. A robot that provides daily services must also perform tasks in a complex dynamic environment, which is a more difficult setting than that of the industrial robot.

Learning by demonstration has proved to be an efficient way for robots to learn new tasks. Using a mapping approach, the robot needs no reprogramming to observe a person's behavior. Thus, this is the most intuitive method for capturing human movement trajectories and mapping them to the robot. However, human and robotic arms have different mechanical structures. Accordingly, the kinematic model of a robot differs from that of a human. Hence, sensory information must be mapped onto the movement space of the devices. A number of kinematic mapping methods have been proposed, including position mapping[1], joint angle mapping[2], and pose mapping[3]. The authors of Ref. [4] proposed a mapping method that uses locality-preserving projections and kNN regression, which achieved relatively good results. Bócsi et al.[5] used a Procrustes analysis algorithm to resolve linear mappings. An efficient multi-class heterogeneous domain adaptation method has also been proposed[6]. However, the calculations of these methods are typically slow. Furthermore, position control in robotic movement planning takes a lot of time to generate the inverse kinematics solution and to perform continuous-points tracking[7]. We propose a method in which the robot's movements are directly controlled by the robotic articulation speed. In this way, the delay of the robotic teleoperation system approaches zero.

It is important to recognize that skill learning[8, 9] is

• Bin Fang, Funchun Sun, and Huaping Liu are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China. E-mail: {fangbin, fcsun, hpliu}@mail.tsinghua.edu.cn.

• Haiming Huang is with the College of Information Engineering, Shenzhen University, Shenzhen 518060, China. E-mail: haimhuang@163.com.

• Xiang Wei and Yuanlong Yu are with Fuzhou University, Fuzhou 350108, China. E-mail: weixiang092@163.com; yu.yuanlong@fzu.edu.cn.

∗ To whom correspondence should be addressed.

Manuscript received: 2018-02-09; revised: 2018-04-13; accepted: 2018-04-27.

the key to the whole system. Numerous skill-learning methods have been proposed in recent years. Metzen et al.[10] developed hierarchical and transfer learning methods that enable robots to learn a repertoire of versatile skills. Their work provides a framework for robot learning of human behavior. Robot learning based on human demonstrations employs behavior segmentation methods. There are also methods that employ imitation learning[11, 12] and reinforcement learning[13]. In this study, to learn movement primitives for robotic interaction, we constructed a model based on Dynamic Movement Primitives (DMPs), which provides a generic framework for motor representation based on nonlinear dynamic systems. DMPs can model both discrete and rhythmic movements, and have been successfully applied to a wide range of tasks, including biped locomotion, drumming, and tennis swings. In Ref. [14], Ijspeert et al. presented the trajectory generation method, which uses probabilistic theory to modify DMPs, which is then called probabilistic movement primitives. Herzog et al.[15] proposed a new approach to combine DMPs with Gaussian processes to enable robots to adapt their roles and cooperation behavior depending on their individual knowledge. Here, we present a system in which any recorded movement trajectory can be represented by a set of differential equations. Then, a trajectory of human motion can be expressed with fewer characteristic variables, thereby realizing the learning of motion characteristics. Finally, we provide a movement that has been learned within certain start and end points, and realize skill learning for human-robot interaction using a wearable device.

The remainder of this paper is structured as follows. In Section 2, we describe the design of the proposed skill learning system, which is based on a teleoperation system installed on a wearable device. In Section 3, we propose a skill learning system that employs modified DMPs. In Section 4, we present our experimental results for the skill learning work, and we draw our conclusion in Section 5.

2 System Description

The skill learning system we developed for human-robot interaction utilizes a wearable device. This system consists of two subsystems: a robotic teleoperation system and an imitation learning system, as shown in Fig. 1.

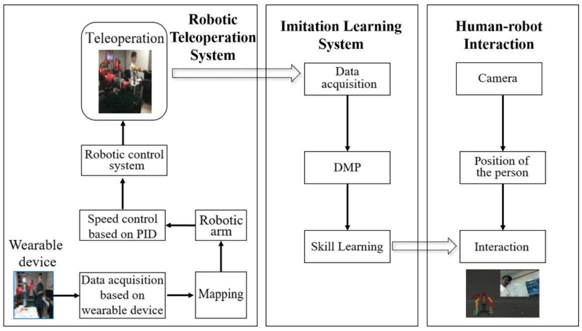


Fig. 1 Proposed skill learning system for human-robot interaction.

2.1 Description of wearable device

The design of the wearable device used for motion capturing is based on Inertial and Magnetic Measurement Units (IMMUs), comprising three-axis micro-machined gyroscopes, three-axis accelerometers, and three-axis magnetometers. In all, there are eighteen IMMUs in the device, which cover all segments of the arm, palm, and fingers. Each string deploys three IMMUs, for a total application of six strings, five of which are used to capture the motions of the five fingers, and the other to capture the motions of the palm, upper arm, and forearm. The battery and microcontroller unit are attached to the wrist. Figure 2 shows the wearable device.

Given the 3D angular velocity, we can estimate the acceleration and magnetic field of a single Inertial Measurement Unit (IMU) and the orientations of the IMMUs by the Quadratic Extended Kalman Filter (QEKF). Then, given the kinematics of the arm-hand, we integrate the constraints to determine their relative orientations. Details regarding the wearable device are presented in Ref. [16].

2.2 Robotic teleoperation system

The Baxter robot that we teleoperated by a human equipped with a wearable device is anthropomorphic. The proposed robotic teleoperation system consists of

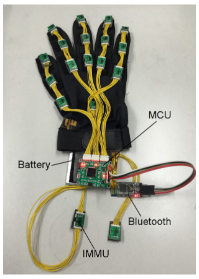


Fig. 2 Wearable device.

three parts, as described in the following.

(1) Data collection

The motion data of the human arm is collected via the wearable device and rends to the computer. Then a node is created in the Robot Operating System (ROS) to send continuous information of the device. Hence the position and attitude of the 18 sensors on the device are collected.

(2) Mapping from wearable device to the robot

In this subsection, we address the problem of mapping from the wearable device to the Baxter robot. This robot has Seven Degrees Of Freedom (7-DOF) and seven force-torque modules that can rotate around their axes. However, although the arm of the human body also has 7-DOF, the kinematic structure of the robot arm differs from that of human beings. For this reason, it is important to construct a map between the movement information of the wearable device and that of the robot arm. Figure 3 shows the structure of the mapping system. The data collected by the wearable device uses the Earth coordinate system, but the movement of the robotic arm uses its own base coordinate system. We utilize a joint-to-joint mapping method[17] for the upper arm. Table 1 shows our proposed mapping structure, which converts the collected data from the device to the 7-DOF robotic system.

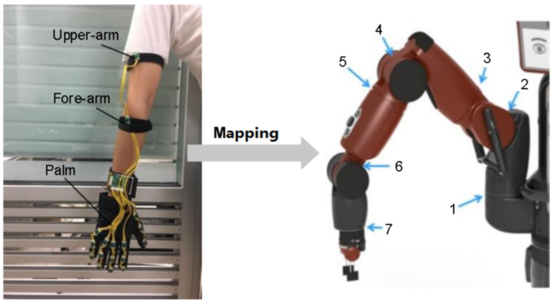


Fig. 3 Teleoperation scheme of 7-DOF robotic system.

Table 1 Mapping commands of 7-DOF robotic teleoperation system.

Human arm	Baxter robotic arm	
	Robotic joint	Data glove
Upper-arm	1	Yaw
	2	Yaw
	3	Yaw
Fore-arm	4	Yaw of upper and fore arm relative angle
	5	Roll of fore and upper arm relative angle
Palm	6	Pitch of palm and fore arm relative angle
	7	Roll of palm and fore arm relative angle

(3) Robotic control

The traditional robotic control system uses the position mode. Although this control method is simple, it has a slow response and other shortcomings. For example, when the target point is determined, the robot can run only at the fixed safe speed to the target point by solving the inverse kinematics, which makes its actions slow and somewhat rigid. We propose a novel method in which we control the speed of each articulated motor of the robot. We use a Proportional-Integral-Derivative (PID) controller[18] to improve the performance of the control system.

$$u(k) = K_p \mathrm{err}(k) + K_I \sum_{j=0}^{k} \mathrm{err}(j) + \\ K_D(\mathrm{err}(k) - \mathrm{err}(k-1)) + u(0) \qquad (1)$$

where $k \in \mathbf{N}$, $u(k)$ is the output value at the k-th sampling, K_p is the scale coefficient, K_I is the integral coefficient, K_D is the differential coefficient, $\mathrm{err}(k)$ is the input deviation at the k-th sampling, and $u(0)$ is the initial value.

Finally, we adjust the speed via speed feedback. In this way, our teleoperation system minimizes the delay by human imitation.

3 Skill Learning Method

Trajectory planning is one of the primary problems in the fine operation of a manipulator. Trajectory planning based on movement primitives requires complex calculations and accurate modeling. In this section, we combine human-teaching trajectory information with DMPs and propose a trajectory learning control method based on DMPs.

3.1 Dynamic movement primitives

Complex motion can be considered as consisting of original-action building blocks executed either in sequence or in parallel. DMPs are mathematical formal expressions of these primitive movements. The difference between a DMP and the previously proposed building block is that each DMP represents a non-linear dynamic system. The basic idea is that you combine a dynamic system that has good rules and stable behavior, and addes another system to it to make it follow some trajectories. DMPs can be divided into two types: discrete and rhythmic. Here we use discrete DMPs. They must be expressed using a convenient and stable dynamic system and one that has a non-linear control term. Here, we use one of the simplest dynamic systems, the damping spring model, in which a damped spring vibrator is used as a skeleton to fit any trajectory. The damping spring oscillator model is as follows:

$$m\frac{\mathrm{d}^2x}{\mathrm{d}t^2} = -kx - c\frac{\mathrm{d}x}{\mathrm{d}t} \tag{2}$$

where m is the mass, k is spring stiffness, and c is viscous damping coefficient.

To simplify the system, this formula is rewritten as follows:

$$\tau\ddot{y} = \alpha_z(\beta_z(g-y)-\dot{y}) \tag{3}$$

where y is the state of our system, τ is a time constant, $\dot{y}$ is the velocity of the joint trajectory, $\ddot{y}$ is the acceleration, g is the final arrival point of the trajectory, and α_z and β_z are gain terms.

Then, we add a forced item f to restrain the repair of our trajectory, as follows:

$$\tau\ddot{y} = \alpha_z(\beta_z(g-y)-\dot{y}) + f \tag{4}$$

After transformation, we can build the conversion system.

$$\begin{aligned} \tau\dot{z} &= \alpha_z(\beta_z(g-y)-z)+f, \\ z &= \tau\dot{y} \end{aligned} \tag{5}$$

The key to the DMP framework is the use of an additional nonlinear system to define the change in the forcing function over time. Here, we import a canonical system, as follows:

$$\dot{x} = -\alpha_x x \tag{6}$$

The forcing function is defined as a canonical system function, which is similar to a radial basis function.

$$f(t) = \frac{\sum_{i=1}^{N}\Psi_i(x)w_i}{\sum_{i=1}^{N}\Psi_i(t)} \tag{7}$$

By importing the normative power system, the forced function is as follows:

$$f(x) = \frac{\sum_{i=1}^{N}\Psi_i(x)w_i}{\sum_{i=1}^{N}\Psi_i(x)}x(g-y_0) \tag{8}$$

where w_i is the weight of each kernel function, and $\Psi_i(x) = \exp(-\frac{1}{2\sigma_i^2}(x-\mu_i)^2)$, where σ_i and μ_i are the width and center point of the basis function, respectively.

In the process of fitting the whole trajectory, a scale attribute is provided. The initial value of x is 1, and becomes 0 as the time extends to infinity. This means that the forced term converges at a point close to the target point.

The robot is teleoperated using the data glove, and a set of motion primitive trajectory learning samples is built. It can directly obtain the position, velocity, and acceleration sequence of each demonstration $(y_{\text{demo}}(t), \dot{y}_{\text{demo}}(t), \ddot{y}_{\text{demo}}(t))$. Then we obtain the initial kinetic equation solution:

$$f_{\text{target}} = \tau^2\ddot{y}_{\text{demo}} - \alpha_z(\beta_z(y_g - y_{\text{demo}}) - \tau\dot{y}_{\text{demo}}) \tag{9}$$

where y_g is the final arrival point of demonstration trajectory.

Finally, we perform a local linear regression to the loss function:

$$J_i = \sum_{t=1}^{P}\Psi_i(t)(f_{\text{target}}(t) - \omega_i\xi(t))^2 \tag{10}$$

where $\xi(t) = x(t)(g-y_0)$.

By optimizing the loss function with locally weighted regression, we obtain the following:

$$w_i = \frac{s^{\mathrm{T}}\Gamma_i f_{\text{target}}}{s^{\mathrm{T}}\Gamma_i s} \tag{11}$$

where $s = \begin{pmatrix} \xi(1) \\ \xi(2) \\ \cdots \\ \xi(P) \end{pmatrix}$, $\Gamma_i = \begin{pmatrix} \psi_i(1) & & 0 \\ & \psi_i(2) & \\ & & \cdots \\ 0 & & \psi_i(P) \end{pmatrix}$,

and $f_{\text{target}} = \begin{bmatrix} f_{\text{target}}(1) \\ f_{\text{target}}(2) \\ \cdots \\ f_{\text{target}}(P) \end{bmatrix}$.

This process can be used with different weights for multiple cores to fit any one track. If a 20-core function is used to represent a trajectory, then the trajectory can be uniquely represented by a 20-dimensional vector.

3.2 Imitation learning system

With the proposed teleoperation system, we designed an imitation learning system based on the DMP. Firstly, we use the teleoperation method of learning. In this approach, the robotic learning action has human action characteristics, unlike other teaching methods. As such, it is a good way to avoid rigidness of the actions, thus making the behavior of the robot more consistent with human behavior norms.

Then we record the movement trajectory of the robotic arm, based on the teleoperation system. Combined with DMPs, we can extract and learn the trajectory-invariant features. This system can learn and reproduce any movement primitives by setting start and end points. In addition, the DMP is preprocessed to achieve rotational invariance, even though it has the characteristics of convergence to the attractor. We record the beginning and end points of the instruction. Then, the system calculates the rotation matrix of the positions of the reproducing and teaching operations.

$$\begin{aligned} P \cdot Q &= |P||Q| \cos \theta \\ \theta &= \arccos\left(\frac{P \cdot Q}{|P||Q|}\right) \end{aligned} \tag{12}$$

where P and Q are the orientation vectors for the starting point and the end point, respectively, and θ is the angle.

Then, we can obtain the rotation matrix:

$$R(\theta) = I + \tilde{\omega} \sin \theta + \tilde{\omega}^2 (1 - \cos \theta) \tag{13}$$

where I is unit matrix, and $\tilde{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$.

Finally, the motion direction and rotation matrix are combined to realize rotation invariance of the DMP. Figure 4 shows a flowchart of the DMP-based skill learning and generation process.

4 Experiments

Based on our previous work, we designed the following experiments to test the skill learning performance. As our experimental platform, we used the standard Robot Operating System (ROS) platform. The system is written in the node form, which perfroms the following roles: a node for wearable device acquisition and analysis, one for data receiving mapping conversion, one for trajectory collection and trajectory execution, and one for trajectory learning generation. We used C++ to implement the main algorithm, and implemented the operation and execution of the robot in Python. In the simulation environment, we used a RealSense camera to achieve human-robot interaction, based on learned movement primitives.

4.1 Robotic teleoperation demonstration

To test the performance of the teleoperation system, one of the authors wore the device and we utilized the Baxter robot. Figure 5 shows our experimental results, in which we can see that by utilizing the proposed speed control mode, the delay of the teleoperation system is nearly zero and the robotic trajectories are smoother than those when using the point control mode.

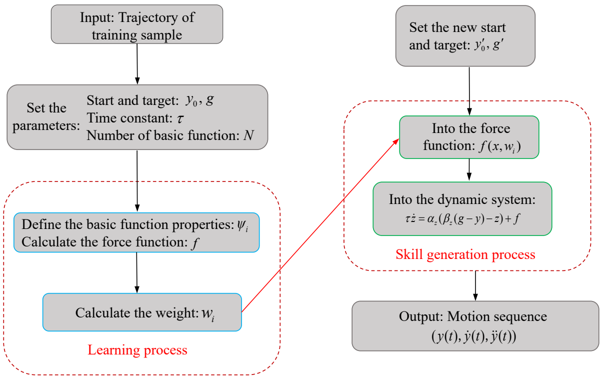

Input: Trajectory of training sample
Set the parameters:
Start and target: y_0, g
Time constant: τ
Number of basic function: N
Define the basic function properties: ψ_i
Calculate the force function: f
Calculate the weight: w_i
Learning process
Set the new start and target: y_0', g'
Into the force function: $f(x, w_i)$
Into the dynamic system: $\tau\dot{z} = \alpha_z(\beta_z(g-y)-z)+f$
Skill generation process
Output: Motion sequence $(y(t), \dot{y}(t), \ddot{y}(t))$

Fig. 4 DMP-based skill learning and generation process

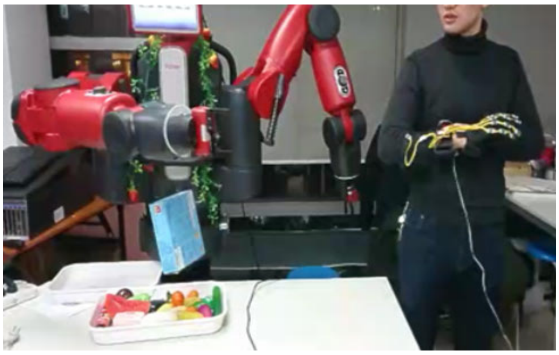
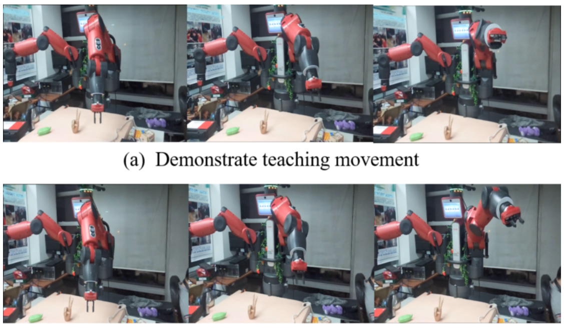


Fig. 5 Teleoperation system.

4.2 Imitation learning experiments

We designed an experiment to verify the performance of the modified DMP. First, we taught the robot a movement and recorded the trajectory by teleoperation, as shown in Fig. 6a. In our previous work, we determined that the robot can reproduce movement trajectories according to the trajectory characteristics of the teaching, as shown in Fig. 6b. Figure 7 shows



(a) Demonstrate teaching movement

(b) Generated movement after study

Fig. 6 Circular movement process based on imitation learning.

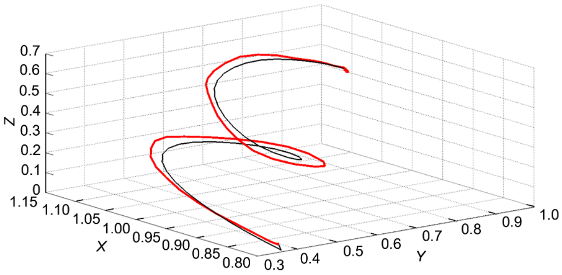


Fig. 7 Analysis of results in which the red curve indicates the original trajectory corresponding to Fig. 6a and the black curve indicates the reproduced trajectories corresponding to Fig. 6b.

an analysis of the data results in which the red curve indicates the original trajectory corresponding to Fig. 6a and the black curve indicates the reproduced trajectories corresponding to Fig. 6b. In the experiments, we used 20 kernel functions to express the trajectory of each dimension.

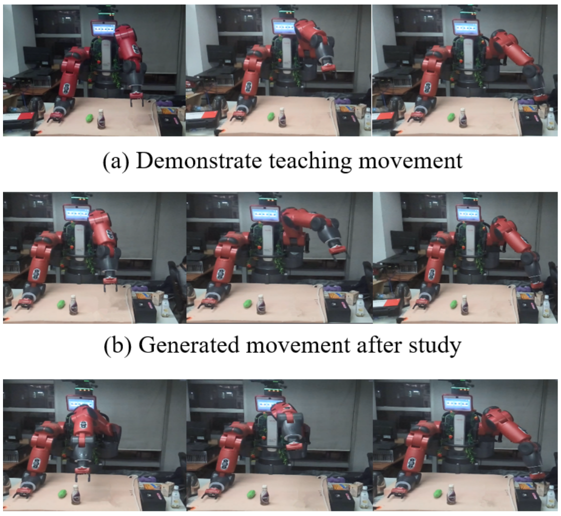
In order to verify the performance of the movement characteristics learning, we designed another experiment in which we changed the initial state to reproduce new trajectories based on the DMP model. Figure 8a shows the original demonstrated movement. Figure 8b shows the generated movement using the learned skill (with the same initial state as that used in Fig. 8a). Figure 8c shows the generated movement with a different initial state. Figure 9 shows the motion trajectories of the three situations. The results confirm the effectiveness of the proposed imitation learning method.

4.3 Establishment of skill-primitive library

In imitation learning, the characterization of movement primitives is the key for gaining robotic skills. Any complicated task consists of smaller tasks, each of which can be split into many sub-actions. Therefore, in order to perform complex tasks, robots must learn many skills before they can intelligently select the required primitive actions. Then we built a small motion-primitive library of dynamic movement primitives, some of which are shown in Fig. 10. To determine the skill generation performance, we extracted the movement primitives and found the generated and original trajectories to be the same. The red line in the figure is the original trajectory, and the blue line is the generated trajectory after robotic learning.

Thus, in our experiments, the robot learned skills from the teaching action via the teleoperation system, and then converted them into their own skills. Using DMPs, our results confirm that we can express complex skill actions in multidimensional matrices of 20 parameters for each dimension, and we can reproduce the applied skills in any situation, thus realizing robotic skill learning. The robot can then use these learned skills in any situation. In addition, after learning the skills, the robot can store them as experience.

To evaluate the overall performance, we then built



(a) Demonstrate teaching movement

(b) Generated movement after study

(c) Generated movement with a different start point

Fig. 8 Imitation learning results for different original states.

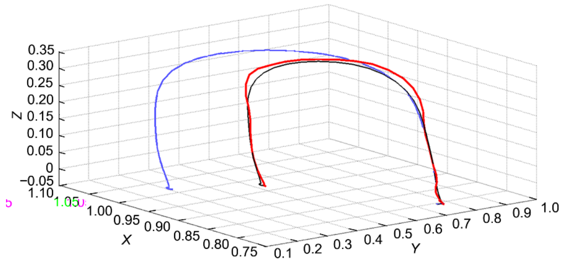
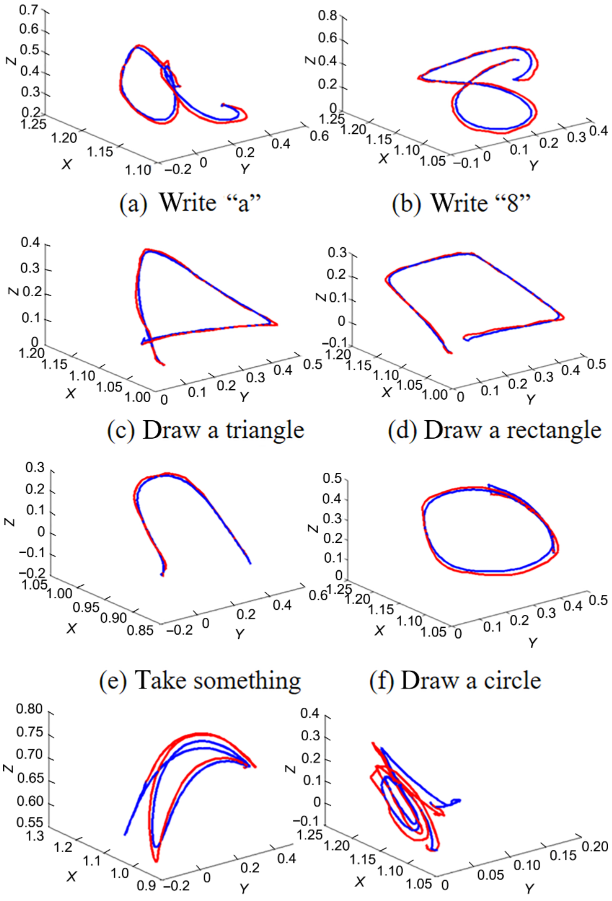


Fig. 9 Motion trajectories of the three situations.



(a) Write “a”

(b) Write “8”

(c) Draw a triangle

(d) Draw a rectangle

(e) Take something

(f) Draw a circle

(g) Greetings

(h) Knocking

Fig. 10 Schematic of movement primitive library.

a set of verification and testing systems based on the completion of the robot learning system. The nature of this learning system is the study of movement primitives, which describe the position of the arm in a time series. Therefore, we used the Dynamic-Time-Warping (DTW)[19] algorithm to calculate the trajectory distance before and after the robotic skill learning. The function of the DTW algorithm is to achieve the minimum overall matching distance between the measured and template eigenvectors, and then to calculate the distance of the optimal matching path with respect to the matching state. Finally, we estimated the degree of similarity between the two traces by the distance between them.

Since the distance between two movement trajectories before and after learning cannot be used to intuitively obtain an expression of their similarity, we designed the trajectory similarity evaluation method (as shown in Algorithm 1) for evaluating the similarity between trajectories.

Using the above process, we can evaluate the learning results of the motivation primitive obtained through skill learning, as stored in the motion primitive library. Table 2 shows the experimental evaluation results.

Algorithm 1 Trajectory similarity evaluation method

```
input:   Template trajectory and test trajectory
output:  Similarity
Step 1:  We sample the two trajectories by percentage area,
         where 50 samples are taken
Step 2:  Calculate the relative distance of sampling points
         in each region and take the average of all distances
Step 3:  Combining with the average of relative distance,
         we select the exponential distribution model to
         represent the similarity model and finally obtain
         the similarity evaluation model: $P = \exp((-1) \times \mathrm{dis}_m)$
```

Table 2 Experimental analysis of movement primitive library.

Movement primitive	Distance (cm)	Similarity (%)
Write “a”	2.2674	98.65
Write “8”	2.2403	98.76
Draw a triangle	1.4883	98.85
Draw a rectangle	1.4765	98.90
Take something	2.2067	98.72
Draw a circle	2.6711	98.36
Greetings	7.3754	95.69
Knocking	3.4330	97.59
Average	2.8948	98.19

The results show the effectiveness of the proposed trajectory learning system, which is developed by the extraction of movement features. After mastering the primitive library skill, the robot can rebuild the trajectory according to the requirement. Table 2 shows that the overall average distance between original movement and generated movement based on DTW is 2.8948 cm, and the average similarity of the skills can reach 98.19%. Therefore, the proposed learning system shows strong learning ability.

4.4 Human-robot interaction demonstration

Lastly, we designed a human-computer interaction experiment in a simulation environment. First, we used a face detection algorithm and a RealSense camera to capture the location of people. Then, the robot made corresponding interaction actions based on the location of the person detected by the camera, as shown in Fig. 11.

5 Conclusion

The teleoperation system we presented in this paper provides an efficient way to transfer human skills to robots. Using this system, we teleoperated the robot movement using a wearable device to directly control the speed of the motors, which effectively reduces the delay time of teleoperation. The proposed method also makes robotic skill learning more efficient. We then proposed an imitation learning system based on the rotation-invariant dynamical-movement primitive method. We performed robotic teleoperation demonstrations and imitation learning experiments, and built a human-robot interaction system, the results of which confirm the effectiveness of the proposed method.

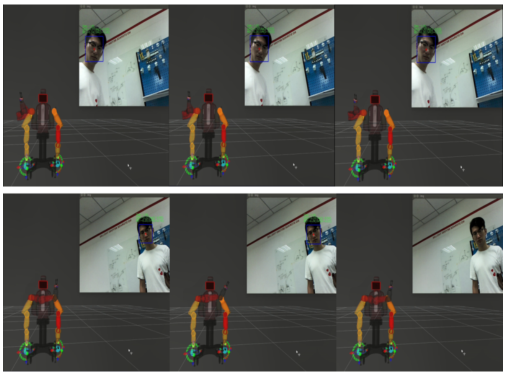


Fig. 11 Human-robot interaction system (each line represents the “hello” action process when the robot detects the person’s position).

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Nos. 61503212, 61473089, U1613212, and 61327809), the Beijing Science and Technology Program (No. Z171100000817007), the German Research Foundation (DFG) in project Cross Modal Learning (No. NSFC 61621136008/DFG TRR-169), and the Suzhou Special Program (No. 2016SZ0219).

References

[1] A. Peer, S. Einenkel, and M. Buss, Multi-fingered telemanipulation-mapping of a human hand to a three-finger gripper, in *Proc. 17th IEEE Int. Symp. on Robot and Human Interactive Communication*, Munich, Germany, 2008, pp. 465–470.

[2] J. Rosell, R. Suárez, C. Rosales, and A. Pérez, Autonomous motion planning of a hand-arm robotic system based on captured human-like hand postures, *Autonom. Rob.*, vol. 31, no. 1, pp. 87–102, 2011.

[3] L. Pao and T. H. Speeter, Transformation of human hand positions for robotic hand control, in *Proc. 1989 Int. Conf. on Robotics and Automation*, Scottsdale, AZ, USA, 1989, pp. 1758–1763.

[4] Y. Lin and Y. Sun, Grasp mapping using locality preserving projections and kNN regression, in *Proc. 2013 IEEE Int. Conf. on Robotics and Automation*, Karlsruhe, Germany, 2013, pp. 1076–1081.

[5] B. Bócsi, L. Csató, and J. Peters, Alignment-based transfer learning for robot models, in *Proc. 2013 Int. Joint Conf. on Neural Networks*, Dallas, TX, USA, 2013, pp. 1–7.

[6] J. T. Zhou, I. W. Tsang, S. J. Pan, and M. K. Tan, Heterogeneous domain adaptation for multiple classes, in *Proc. 17th Int. Conf. on Artificial Intelligence and Statistics*, Reykjavik, Iceland, 2014, pp. 1095–1103.

[7] S. Schaal, Dynamic movement primitives—A framework for motor control in humans and humanoid robotics, in *Adaptive Motion of Animals and Machines*, H. Kimura, K. Tsuchiya, A. Ishiguro, and H. Witte, eds. Springer, 2006, pp. 261–280.

[8] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, Learning and generalization of motor skills by learning from demonstration, in *Proc. 2009 IEEE Int. Conf. on Robotics and Automation*, Kobe, Japan, 2009, pp. 763–768.

[9] 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.

[10] J. H. Metzen, A. Fabisch, L. Senger, J. de G. Fernández, and E. A. Kirchner, Towards learning of generic skills for robotic manipulation, *Künstl. Intell.*, vol. 28, no. 1, pp. 15–20, 2014.

[11] T. H. Yu, C. Finn, A. N. Xie, S. Dasari, T. H. Zhang, P. Abbeel, and S. Levine, One-shot imitation from observing humans via domain-adaptive meta-learning, arXiv preprint arXiv: 1802.01557, 2018.

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

[13] J. Kober, J. Bagnel, and J. Peters, Reinforcement learning in robotics: A survey, *Int. J. Rob. Res.*, vol. 32, no. 11, pp. 1238–1274, 2013.

[14] A. J. Ijspeert, J. Nakanishi, and S. Schaal, Movement imitation with nonlinear dynamical systems in humanoid robots, in *Proc. 2002 IEEE Int. Conf. on Robotics and Automation*, Washington, DC, USA, 2002, pp. 1398–1403.

[15] S. Herzog, F. Wörgötter, and T. Kulvicius, Optimal trajectory generation for generalization of discrete movements with boundary conditions, in *Proc. 2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Daejeon, Korea, 2016, pp. 3143–3149.

[16] B. Fang, F. C. Sun, H. P. Liu, and D. Guo, Development of a wearable device for motion capturing based on magnetic and inertial measurement units, *Scientific Programming*, vol. 2017, p. 7594763, 2017.

[17] B. Fang, F. C. Sun, H. P. Liu, D. Guo, W. D. Chen, and G. D. Yao, Robotic teleoperation systems using a wearable multimodal fusion device, *Int. J. Adv. Rob. Syst.*, vol. 14, no. 4, pp. 1–11, 2017.

[18] K. H. Ang, G. Chong, and Y. Li, PID control system analysis, design, and technology, *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 4, pp. 559–576, 2005.

[19] A. Vakanski, I. Mantegh, A. Irish, and F. Janabi-Sharifi, Trajectory learning for robot programming by demonstration using hidden markov model and dynamic time warping, *IEEE Trans. Syst., Man, Cybern. B: Cybern.*, vol. 42, no. 4, pp. 1039–1052, 2012.

Bin Fang received the PhD degree in mechanical engineering from Beihang University in 2014, and now he is a research assistant in the Department of Computer Science and Technology at Tsinghua University. His research interests include sensor fusion, wearable devices, and robotics and human-robot interaction.

Xiang Wei received the BS degree from Shenyang Aerospace University, Shenyang, China, in 2014. He is pursuing the MS degree at the Fuzhou University, Fuzhou, China. His current research interests include experience learning and machine learning.

Fuchun Sun received the PhD degree in computer science from Tsinghua University in 1997. He is a full professor with the Department of Computer Science and Technology, Tsinghua University, China. His current research interest includes robotic perception and cognition. He was a recipient of the National Science Fund for Distinguished Young Scholars. He serves as an associate editor of a series of international journals including *IEEE Transactions on Systems, Man and Cybernetics: Systems*, *IEEE Transactions on Fuzzy Systems, Mechatronics*, and *Robotics and Autonomous Systems*. He is a fellow of the IEEE.

Yuanlong Yu received the PhD degree from Memorial University of Newfoundland, Canada, in 2010. Since 2011, he has been a post-Doctoral fellow with Memorial University of Newfoundland and Dalhousie University. Since 2013, he has been a professor with Fuzhou University. His current research interests include computer vision, machine learning, visual attention, autonomous mental development, and cognitive robotics.

Huaping Liu received the PhD degree in computer science from Tsinghua University in 2004. He is an associate professor with the Department of Computer Science and Technology, Tsinghua University. His current research interests include robotic perception and learning. He serves as an associate editor of some journals including *Cognitive Computation, Neurocomputing, IEEE Transactions on Industrial Informatics*, *IEEE Transactions on Automation Science and Engineering*, and *IEEE Robotics and Automation Letters*, and some conferences including ICRA and IROS. He also served as the program committee member of RSS, IJCAI, and AAAI.

Haiming Huang received the PhD degree from Beihang University in 2016. He is currently a postdoctoral in the College of Information Engineering, Shenzhen University. His research interests include soft robotics, flexible sensor, embedded mechatronics control, and robotics.