Received May 3, 2021, accepted May 9, 2021, date of publication May 17, 2021, date of current version July 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3080838

A Skill Programming Method Based on Assembly Motion Primitive for Modular Assembly System

PENGYUE GUO[1], ZHIJING ZHANG[1], YAN LIU[1], YUJUN LIU[1], DONGSHENG ZHU[1], AND CHAO SHAO[1,2]

[1]School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China
[2]Department of Mechatronics, Beijing Institute of Control Engineering, Beijing 200085, China

Corresponding author: Zhijing Zhang (zhzhj@bit.edu.cn)

This work was supported in part by the National Key Technology Research and Development Program of China under Grant SQ2019YFB130212, and in part by the National Defense Basic Scientific Research Project of Science and Industry Bureau of China under Grant JCKY2016208B004.

ABSTRACT To improve the programming efficiency of automatic assembly system, a novel skill programming framework based on task learning is proposed for modular assembly system in this paper. In this framework, the motion sequence of assembly skills can be modeled by demonstration data. And the assembly task is represented hierarchically. A complete assembly process of a part is divided into several skills, and each skill is divided into several sequential assembly motion primitives (AMP) of multiple modules. Then, a learning method of assembly motion sequence based on Hidden Markov Model is proposed, and the maximum probability method is used to generate the optimal sequential AMP. Each AMP is input to the assembly system in the form of instruction to complete the assembly. Aiming at the problem of accurate positioning and trajectory planning, visual guidance and direct teaching method are used to settle this problem. To evaluate the viability of the proposed framework, a customized modular assembly system is used to acquire the demonstration data, and a graphical user interface (GUI) software is designed. Five assembly skills are learned. Experimental are conducted to validate the effectiveness of the proposed method.

INDEX TERMS Assembly motion primitive, modular assembly system, hidden Markov model, skill learning.

I. INTRODUCTION

With the development of automatic assembly for custom-made and low-volume, the automatic assembly system needs to have the ability to transfer from one task to another efficiently [1]. Because different types of sensors have their own limitations, the modular assembly system integrating multiple sensors can meet the requirements of complex assembly tasks. In addition to trajectory teaching, the modular assembly system also has functions such as image processing [2], force feedback control [3], IO control, multi-modular parallel or serial control, and collision detection [4]. And the modular assembly system has the characteristics of multi-information fusion, complex coordination, and diverse control methods [5]. In the traditional programming of automatic assembly process, writing computer code and the use of teach pendants are the predominant method. The programming experts first program according to the previous programming expertise, and refine the assembly process through trials-and-error. It needs tedious reprogramming to adapt to new assembly tasks. Therefore, the existing programming mode is difficult to quickly realize the programming of a large number of assembly process for new tasks. The offline programming needs complete and accurate design model, task experts on the factory floor and development engineers to complete.

The recent development of data-driven artificial intelligence, some learning-based methods are highly improving the efficiency of programming. In the industrial scene, a large number of methods based on learning from demonstration (LFD) or programming by demonstration (PBD) are widely used in task learning and motion learning [6]. In the field of robotics, PBD provides a new way to transfer knowledge from human to robot, which is an important way to simplify programming. And PBD provides a skill programming method without professional computer language experts,

The associate editor coordinating the review of this manuscript and approving it for publication was Gustavo Olague.

which transfers low-level mechanical control to high-level skill representation control.

Inspired by the thinking of human programming, the experience data of programming can be modeled as a nonlinear model, and decision-making is carried out like the brain. For the data extraction and mathematical representation of automatic assembly process of various parts, there are few related theories and models. The assembly process data is a discrete motion sequence, and contains a variety of sensor information, it has the characteristics of nonlinear, time-varying, uncertainty, and so on. Hidden Markov model (HMM) [7] is a kind of probability and statistics model about time series. It can represent all training data in the model through its parameters, which enables us to obtain the most likely skill description of human demonstrated data. For example, in the grasping skill, the manipulator first moves to the top of the parts through path planning because parts are stored in the grooved tray. Then the manipulators move vertically downward to grasp. Afterwards, the manipulators lift the part and transport the parts to the next position. At the same time, in order to improve efficiency, other modules of the system perform independent assembly motion or assistance motion. The modular assembly system needs a very interweaved assembly sequence to complete the assembly skills, which is different from the robot system.

To autonomously execute the above-mentioned tasks, the motivation of this article is to design and implement a rapid programming framework that can be learned from demonstration. This means the modular assembly system needs to have the ability to adapt to new tasks, and then it can map the demonstration data into mathematical model. Instruction and PBD are combined in our work to express and learn high-level task skills. This framework provides an accessible method for non-experts to quickly and easily program. The main contributions of this work are as follows.

(1) Modular and reconfigurable assembly components which integrates multiple sensors are designed.

(2) An autonomous modular software system was developed, which can be programmed via script commands.

(3) A skill learning framework is proposed which expresses the assembly skills as learned HMMs. The assembly action primitive (AMP) is proposed in this article for the first time. The assembly process is decomposed into AMPs, which are expressed by instructions. The demonstration assembly sequence is used as the training data of HMM.

(4) Some innovation experiments were shown for the first time. The assembly action data of various parts is extracted based on the modular assembly system. The proposed method is compared with other learning methods, and the method is applied to the actual assembly of new parts.

In this work, 12 kinds of motion instructions are recorded. And 5 kinds of skill learning are completed, which are grasping, handover, alignment, and two kinds of insertion skills. The generated optimal motion sequence is input into the modular assembly system as a skill. The proposed methods were evaluated through the implementation of multiple, real-life industrial production processes involving modular assembly system.

The organizational structure of this article is as follows: In Section II, the related works are reviewed. In Section III, The configuration of the modular assembly system is presented. In Section IV, the skill learning framework and the general motion learning scheme is proposed; In Section V, the mathematical model of this paper is introduced; In section VI, the experimental results of the new framework are revealed. Finally, the conclusions are given in Section VII.

II. RELATED WORK

In assembly scenario, PBD can be divided into motion-based (low-level) PBD and task-based (high-level) PBD [8]. The low-level representation of the skill, taking the form of a non-linear mapping between sensory and motor information. The high-level representation of the skill that decomposes the skill in a sequence of motion units, which we will refer to as symbolic encoding.

A. MOTION-ORIENTED PROGRAMMING BY DEMONSTRATION

In order to quickly realize the programming of the assembly process, many PBD methods of assembly motions have been produced. According to the data acquisition method, PBD can be divided into observation learning, kinesthetic teaching and teleoperation. The learning method of obtaining demonstration data through computer vision is currently more popular [9]–[11]. These approaches had completed the actions of grasping, moving, and screwing into the hole. However, these methods are not suitable for scenarios requiring high assembly accuracy and systems with compact structures. On the other hand, kinesthetic teaching and data glove teleoperation [12]–[14] provide an feasible method for non-expert to quickly and easily program robots. A data glove and force sensors are used to collect teaching data. Most of these methods aimed at achieving path optimization and local replanning. Nevertheless, these methods cannot deal with the multi-modular movement and are not sufficient to describe the semantics in the task level.

In the aspect of motion trajectory planning and collision detection, many motion-oriented studies have come into play. Traditionally, dynamic movement primitives (DMPs) [15] are represented low-level motion primitives which can be stored in a library. And then DMPs are called by high-level planner. Though the use of Gaussian mixture model (GMM) [16], the statistical characteristics of the set of trajectories can be extracted. The robot can reuse the learned skills to handle new tasks. In the recent years, a new type of admittance-based control method [17] is used to generate collision avoidance trajectories, which can be effectively applied to physical human-robot interaction systems. Another [18] trajectory generation method is based on imitation learning, which designs a neural network-based strategy module to infer the desired motion in the image space. Additionally, some hybrid methods have been proposed. In [12], HSMM-GMR was used to handle multimodal learning with position and force constraints, and showed good performance. The above studies had shown that the control strategy that decomposes movement into a series of motion primitives is feasible. However, the above methods focus on the single robot that moves freely, the existing learning programming can only complete the simple position and trajectory learning, so it is difficult to apply to a compact multi-module assembly system.

For precision assembly of micro and small parts, it is difficult to meet the high-precision assembly requirements with a single robot guidance. Multi-module and multi-sensor integration can accomplish such tasks. In order to implement the assembly skills of grasp, handover, alignment, and insertion, a multi-modal control method is proposed in this paper. In grasping skill, to obtain the desire position of the part, the visual guidance method [19] is adopted. For obstacle avoidance in the movement process, the method of adding avoidance points is presented. In handover skill, the desired position is recorded through the demonstration of the teach pendant. In insert skills, a compliance controller is designed.

B. TASK-ORIENTED PROGRAMMING BY DEMONSTRATION

In task learning, the task is encoded according to a predefined sequence of motion elements. A series of encoding sequences are regarded as assembly skills. It is necessary to understand and reason about the acquired information, and can apply it to the similar task. According to different types of tasks, task-oriented programming by demonstration has various representation methods. In recent years, task-oriented PBD methods based on reinforcement learning (RL) [20] have received more attention. This method offers a new paradigm for acquiring skill by maximizing the overall reward. But this method requires a lot of demonstration data and is only suitable for simple tasks. In [21], a closed-loop learning method through demonstration, feedback, and transfer is proposed. The GUI is designed to allow users to optimize the knowledge of high-level tasks and low-level tasks. To solve the manipulation task, a task planning and execution framework [22] oriented to service robots was designed, which completed the deployment of multiple skills through empirical knowledge modules. In task learning, the task can be coded according to a pre-defined sequence of motion elements, and a series of coding sequences can be regarded as assembly skills. In [23], a syntactic approach was described that captures important task structures in the form of probabilistic activity grammars from a reasonably small number of samples. Bernardin *et al.* [24] presented a symbolic and semantic representations approach which stores the sequence of activities in a directed task graph. The new task process is generated by reasoning. In [25], an assembly sequence learning method based on demonstration programming and active exploration is proposed. This method studies the learning problems of priority constraints, object relative size and position constraints in automatic assembly planning. The task of inserting shaft holes with various shapes in the method is finished. Unfortunately, the above-mentioned semantic-based learning methods require a complicated modeling process for complex tasks and new tasks.

To adapt to the changes of environments, an assembly sequence classification algorithm [26] based on a hierarchical bidirectional Long Short Term Memory network is proposed, which divides the motion data into therbligs. This method is used to represent 7 common tasks successfully. Sasabuchi *et al.* [27] proposed a complex task sequence learning method combining motion knowledge and task knowledge. This method integrated both task constraints of each individual task and human motion mimicking and completed the task of opening the refrigerator and picking objects. However, the above methods also require a large amount of demonstration data, and cannot address the problem of multi-modal motion sequence encoding. Because it is unrealistic and expensive to obtain a large number of skill data of automatic assembly [28], the HMM based on a small number of samples training is feasible. In terms of data modeling for motion-oriented PBD, Hidden Markov Model is a general method of processing time series data in the field of robot PBD [29]–[31]. By describing the motions as a sequence of observations corresponding to a series of hidden state sequences, this method can have a good accuracy rate when the amount of data is small.

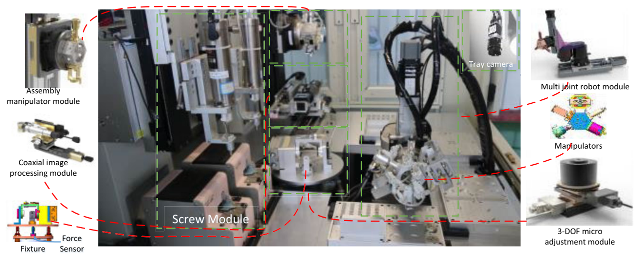


FIGURE 1. Modular assembly system.

III. OVERVIEW OF MODELAR ASSEMBLY SYATEM

A modular assembly system that can assemble a variety of parts is shown in Fig. 1, consisting of a multi-joint robot module (MJRM), an assembly execution manipulator module (AEMM), a coaxial processing module (CAM), a 3 degree-of-freedom (DOF) micro-motion adjustment module (MAM), a screwdriver module, and a clamping module. The system has multiple sensors and controllers.

The MJRM consists of one translational stage, one vertical lifting stage, and three rotational stages. The MJRM is driven by stepped motors. The non-coupling of linear motion and rotation can effectively reduce movement errors when moving in a certain direction.

The AEMM consists of a vertical motion axis with high straightness and a rotation axis. A manipulator is equipped with on the rotation axis, which can grip various parts.

The CAM is a high-precision alignment system [32] developed by our research team. It consists of two horizontal axes, a camera, two light sources, and a prism. The target part and the basal part are imaged on the same image plane, which reduces the calibration error.

The 3-DOF MAM includes linear movement along the X and Y directions and rotation around Z. The control method adopts a closed-loop mode, and error compensation is performed to make it have micrometer-level positioning accuracy. The force sensor is installed on the module.

The screwdriver can automatically absorb and screw two types of small screws through input/output (IO) signal control.

The manipulators can realize the grasping and releasing of various parts though IO signal control.

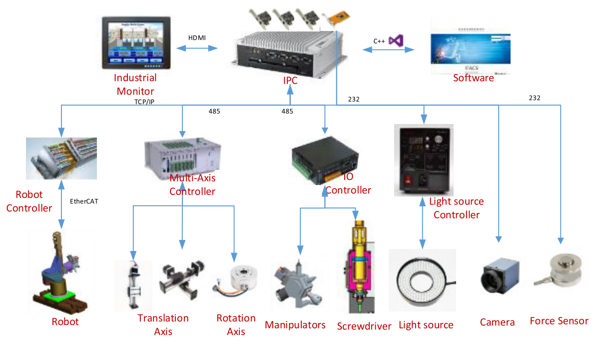


FIGURE 2. System hardware setup.

The modular assembly system has a manual-operation interface on the host computer, which provides the image display, the system state display, the instruction parse, and the manual-control buttons. The connection and control methods of each module are shown in the Fig. 2. The template matching algorithm in the software development kit Mil developed by Matrox is used to calculate the pose of the part. The integrated control of the system is completed on an industrial computer. The software of control system is programmed through C++. And the software is also a software platform that can write action sequences instructions. A series of instructions are written to complete the entire assembly task. For the control of each axis, we adopt the traditional PID control method.

IV. ASSEMBLY SKILL PROGRAMMING FRAMEWORK

A. TASK DESCRIPTION

For assembly tasks, the constraint of the assembly object in this article is that the part moves from top to down. In addition, we assume that the parts to be assembled have some pre-defined task constraint. Such as, assembly skills set is generated according to the attributes of the parts (i.e. first to grab, second to align, final to insert). In addition to the skill constraints of assembly, a certain skill also contains motion constraints of a multi-step sequence. For example, in the grasping skill, the manipulator must move directly above the part before grabbing the part. And the MJRM needs to pass through the obstacle avoidance points before transporting the parts to the assembly area. However, some motion sequences are random, such as the single-axis motion sequence in a certain module. Although the skill constraints are predefined, the motion constraints are not explained and must be extracted from the demonstration. The designed framework uses the entire system state for compact movement learning.

FIGURE 3. Example of parts assembly.

Fig. 3 shows an example application that has our problem setting. Below we explain in more detail the motion constraints of a certain skill in the modular assembly system.

1) THE GRASPING SKILL

As mentioned above, the tray camera obtains the pose of the part through the template matching algorithm. Then the grasping position of MJRM is calculated. The MJRM and the manipulators move to the position of the part, and grab the part by clamping or adsorption. Then the MJRM transports the parts to the assembly area. Meanwhile, the 3-DOF MAM and the AEMM move to the desired position to prepare for the next assembly skill.

2) THE HANDOVER SKILL

Because the AEMM has higher motion accuracy than the MJRM, the parts are handed over to the AEMM from MJRM.

3) THE ALIGNMENT SKILL

To ensure that the basal parts and target parts have good assembly accuracy, the CAM is used to align the positions of the two parts. First, the CAM is move to the assembly area. Second, the upper light source is opened, and the image of the target part is captured. Similarly, the lower light source is opened, and the image of basal part is also captured. The template matching algorithm is used to calculate the pose error of the two parts. Third, the 3-DOF MAM is adjusted based on the above results.

4) THE INSERT1 SKILL

When the assembly accuracy of the two parts is relatively low, the MJRM controlled the manipulator move the target part downward. Then the manipulator releases the part and then leaves.

5) THE INSERT2 SKILL

The second insert skill is that the AEMM controlled the manipulator move the target part downward, meanwhile adjusting its vertical position to keep the contact force within the allowed range. Similarly, the manipulator releases the part and then leaves.

B. ASSEMBLY MOTION PRIMITIVE

Previous research shows that decomposing a motion sequence into a series of motion primitives can lead to a positive impact on designing the control strategy and analyzing the motion sequence [33]. The assembly motion primitive is the basic element of a series of assembly actions. It is a way to classify and express various assembly actions. In this paper, the AMPs includes eight action instructions and 12 action numbers, as shown in Table. 1.

TABLE 1. Description of assembly motion primitive.

Motion Number	Assembly Motion Primitive	Descriptions
1	RobotMove	Move the robot
2	AxisMove(1,2,3)	Move the axis 1,2, or 3 (3-DOF MAM axis)
3	AxisMove(5,6)	Move the axis 5 or 6 (AEMM axis)
4	AxisMove(4,7)	Move the axis 4 or 7 (CAM axis)
5	IO	Trigger Input/Output signal of Manipulator
6	IO	Trigger Input/Output signal of fixture
7	IO	Trigger Input/Output signal of AEMM
8	LighSource	Adjust the intensity of the two light sources
9	ImageProcessing	Execute the image processing program and output the result
10	AdjustWithResult	Adjust the 3-DOF adjustment module based on the results of image processing
11	ForceControlAxis	Move the axis based on force feedback
12	RobotAdjusting	Move the robot based on the results of image processing

In the above primitive, the axes and IOs of each module are individually numbered. Each Skill is composed of a set of AMPs. The speed and acceleration of axes is set before assembly. The relationship between module and primitive is shown in Fig. 4. Such as the instruction of "*RobotMove*", the control object of this instruction is the MJRM. The instruction of "*IO*", the control object of this instruction is the manipulators, fixture, and the screwdrivers.

For the multi-sensing modular automated assembly system, it is still at the stage of special planes, and it is impossible to expand the application scenarios. Therefore, similar

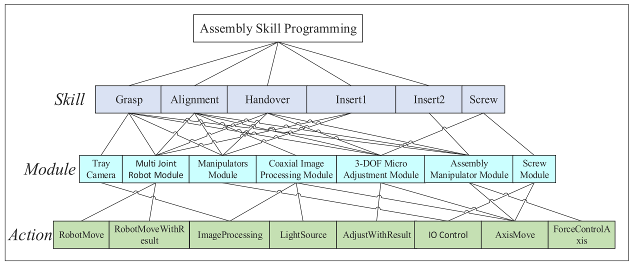


FIGURE 4. Hierarchical representation of assembly skills.

to the robot programming language, this article develops a set of easy-to-understand natural language-like assembly instruction set and GUI. This type of instruction set integrates instructions such as visual feedback, force feedback, axis motion, IO response, light source control, status query, and robot movement. The editing interface of the instruction is shown in Fig. 5. The system software can parse each instruction and complete the corresponding action.

C. ASSEMBLY ACTION GENERATION

Aiming at the above-mentioned complex programming problem of automatic assembly of parts, a rapid programming method of assembly action sequence based on HMM is proposed. For a given assembly task, the assembly skill is regarded as a limited sequence of actions. Since the process of part assembly involves multi-sensor sensing and path planning, the strategy adopted in this paper is to first generate an effective action sequence through the HMM, and the precise positions and parameters contained in the action sequence are obtained through direct teaching. The goal of skill learning is to establish the most probable performance model from all recorded data and generate an AMP sequence that is closest to the most probable performance. Here, we regard the assembly skill instruction as an observable random process, and the knowledge or strategy behind it is an implicit random process.

The strategy of this article is to learn assembly skills through demonstration. After manually determining the assembly skills required for new parts, the proposed method automatically generates multi-step AMP for each skill. Then the script programming work is completed based on the above AMP and external input parameters. The overall process of this part of work is shown in the Fig. 5.

V. METHODOLOGY

For a given task, human demonstrations represent the skills and intentions, which can be coded by multiple hidden Markov models. The hidden Markov model is used to model the probabilistic transition between discrete. The optimal action sequence is obtained by the maximum probability method and used as the action input of the new task, thereby simplifying complicated manual programming tasks.

A. ASSEMBLY MOTION SEQUENCE REPRESENTATION

In the field of precision assembly, grasping, aligning, handover, and inserting [1], [9], [29], [30] are common

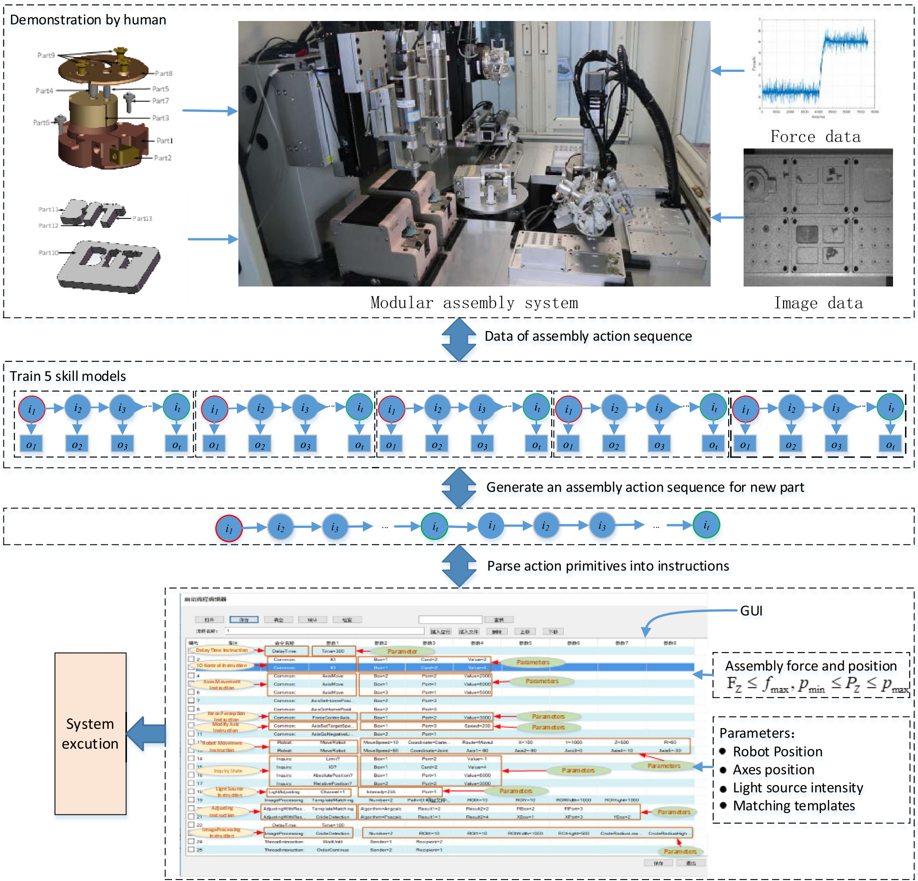


FIGURE 5. Learning framework of assembly skills.

assembly skills. Hidden Markov model is used to learn the above 5 skills. Consider a skill which can be described at any time as being in one of a set of N distinct states $S_1, S_2 \ldots, S_N$, and the states are unobservable. The actual at time t measured from observation is denoted by i_t. When the skill is in state $i_t = S_t$, M distinct output symbols $V_1, V_2, \ldots, V_M$ can be observed. For each skill, there is a state sequence $I = (i_1, i_2, \ldots, i_T)$ of length T and a corresponding observation instruction sequence $O = (o_1, o_2, \ldots, o_T)$. $A = [a_{ij}]$ represents the state transition matrix, and $B = [b_j(k)]$ represents the observation probability matrix.

At time $t+1$, the skill goes to state $i_{t+1} = S_j$ with transition probability a_{ij}, and $\sum_j (a_{ij}) = 1$, where

$$a_{ij} = P[i_{t+1} = S_j | i_t = S_i], \quad 1 \leq i, j \leq N \tag{1}$$

B is associated with each state, and $\sum_k b_j(k) = 1$,where

$$b_j(k) = P(o_t = V_k | i_t = S_j), \quad j = 1, 2, \ldots, N \tag{2}$$

Here, π is used to represent the probability vector of the initial state. Therefore, a unique and definite hidden Markov model is obtained. In order to evaluate the parameters of the model, we need to obtain detailed action sequence data. For a given skill, the human intention can be expressed by the trained model. Assume that the skill contains m sub-actions and uses n states from left to right.

B. LEARNING SKILL THOUGH HMM

Since the current automated assembly of parts is still in the exploratory stage, there is no publicly available automated assembly data set. The data in this article is obtained by extracting the automated action sequence of a variety of parts in the modular assembly system, which is a series of discrete data points. Using a multiple HMM, learning is achieved by adjusting the model parameters to maximize the probability of the observation sequence. $\lambda = \{A, B, \pi\}$ is the complete parameter set of the HMM.

The HMM is trained by S assembly action sequences of length T. This paper uses an unsupervised learning method to solve the HMM model parameters. One action sequence is D observation sequence $\{(O_1, I_1), (O_2, I_2), \ldots, (O_D, I_D)\}$ with length T. The unknown hidden state sequence corresponding to observation sequences $O_d = \{o_1^d, o_2^d, \ldots, o_T^d\}$ is expressed as $I_d = \{i_1^d, i_2^d, \ldots, i_T^d\}$. An iterative algorithm is used to update the model parameters. Consider any model λ with non-zero parameters. We first define the posterior probability of transitions $\xi_t(i, j)$, from state i to state j, given the model and the observation sequence,

$$\xi_t(i,j) = P(S_t = i, S_{t+1} = j|O,\lambda) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \tag{3}$$

where $\alpha_t(i) = P(O_1, O_2, \ldots, O_t, S_t = i|\lambda)$ is defined as the forward variable, a backward variable can be defined as $\beta_t(i) = P(O_{t+1}, O_{t+2}, \ldots, O_T|S_t = i, \lambda)$.

Similarly, the posterior probability of being in state i at the time t, $\gamma_t(i)$, given the observation sequence and model, is defined as

$$\gamma_t(i) = P(S_t = i|O,\lambda) = \frac{\alpha_t(i)\beta_t(j)}{\sum_{k=1}^{N}\alpha_t(i)\beta_t(j)} \tag{4}$$

The Baum-Welch [34] algorithm can be extended to the case based on the signal independence assumption. We can establish the following formula to re-estimate the HMM parameters:

$$\pi_i = \frac{\sum_{d=1}^{D}\gamma_1(i)}{D} \tag{5}$$

$$a_{ij} = \frac{\sum_{d=1}^{D}\sum_{t=1}^{T-1}\xi_t^{(d)}(i,j)}{\sum_{d=1}^{D}\sum_{t=1}^{T-1}\gamma_1^{(d)}(i)} \tag{6}$$

$$b_j(k) = \frac{\sum_{d=1}^{D}\sum_{t=1,o_t^{(d)}=V_k}^{T-1}\gamma_t^{(d)}(i)}{\sum_{d=1}^{D}\sum_{t=1}^{T-1}\gamma_t^{(d)}(j)} \tag{7}$$

The model parameters are adjusted in such a way that they can maximize the likelihood $P(O|\lambda)$ for the given set of training data. The trained HMM represents the model of the best trajectory.

C. GENERATION OF ASSEMBLY ACTION SEQUENCE

Having determined the model representing the most likely human performance, we now look for the time sequence which best matches the trained model. The generation function decodes motion sequence from the HMM. Then outputs probability that the data is generated by the HMM.

$$\phi_t = \arg\max_{1\le k\le M} P(o_t = V_k|i_t = S_i) \tag{8}$$

$$\Psi_{t+1} = \arg\max_{1\le j\le N} P(i_{t+1} = s_j|i_t = S_i) \tag{9}$$

where Ψ_{t+1} represents the next state with the maximum probability in the ith state of the model, ϕ_t represents the action data O with the maximum probability of output in the ith state under the model. Suppose the number of actions for a certain skill is n, and finally a specified number (n) of skill actions are generated as

$$M_n = \{motion(\eta), \eta = 0, 1, \ldots, n\} \tag{10}$$

VI. EXPERIMENTS AND RESULTS

A. SKILL LEARNING EXPERIMENT

In this section, the experiment was built based on the design system in Section III. The software system allows flexibly modification of action instructions according to the different parts to be assembled, and can also output the action sequence data demonstrated by humans in the form of text. In this demonstration experiment, the modules of the assembly system cooperated to complete the assembly tasks of 13 kinds of parts. Firstly, according to the characteristics of the parts, the assembly tasks were performed by experienced system engineers. Such as complex position teaching, image processing, and assembly process writing. And the automated assembly experiment was completed on the modular assembly system. Then, the successful action sequence was artificially expressed numerically and segmented. The principle of numerical representation is to use numbers to represent the action instructions. The representation rules are shown in Table 1. The segmentation of assembly task is composed of two parts. First, the entire assembly sequence is segmented according to different parts. Second, the assembly action sequence of each part is represented by five skills. The 13 kinds of parts are shown in Fig. 6(a) and (b). After skill segmentation and numerical representation, the result is shown in the Fig. 7. The assembly skills of the 13 types of parts assembly are as follows. The assembly skills of parts 1, 2, 3, 6, 7, and 8 are {grasp, handover, alignment, insert1}, and part 4 is {grasp, handover, insert2}, part 5 is {grasp, insert2), part 9 and 10 are {grasp, insert1}, parts 11, 12, and 13 are {grasp, alignment, insert1}. The demonstration process is shown as Fig. 3.

By using the method proposed in section V of this article to learn the parameters of the HMM model, the state transition

$$A = \begin{bmatrix} 0.002 & 0.998 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.021 & 0.979 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.441 & 0.559 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.255 & 0.745 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.241 & 0.789 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.259 & 0.741 & 0 & 0 & 00 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.035 & 0.965 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.355 & 0.645 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.145 & 0.855 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.016 & 0.984 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

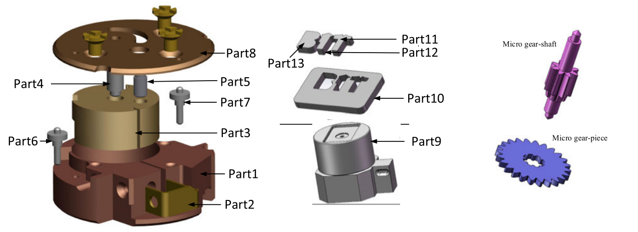


FIGURE 6. Parts to be assembled.

TABLE 2. The observation probability matrix of grasp skill.

State	1	2	3	5	6	7	8	12
1	0	0.1	0	0	0.101	0	0	0.799
2	0	0.299	0	0.505	0.098	0.098	0	0
3	0.146	0.585	0	0.269	0	0	0	0
4	0.2	0.569	0.228	0.002	0	0	0	0
5	0.032	0.26	0.708	0	0	0	0	0
6	0.548	0.269	0.133	0.05	0	0	0	0
7	0.001	0	0.107	0.886	0	0	0	0
8	0.932	0	0.017	0.051	0	0	0	0
9	0.803	0	0	0.197	0	0	0	0
10	0.612	0	0	0.005	0	0.383	0	0
11	0.065	0	0	0.935	0	0	0	0

matrix of grasping skill is obtained. Among them, the initial parameters π of training are set to 0.4, 0.2, 0.2, 0.2, 0, 0, 0, 0, 0, 0, 0, matrix A, as shown at the bottom of the previous page.

For each state, we get the probabilities of the commanded action from the observation symbol probability matrix B. B is shown in Table. 2.

From these parameters we can easily find the most likely action sequence based on the proposed method in section V. The generated grasping skill sequence containing 11 actions is $12 \rightarrow 5 \rightarrow 2 \rightarrow 2 \rightarrow 3 \rightarrow 1 \rightarrow 5 \rightarrow 1 \rightarrow 1 \rightarrow 1 \rightarrow 5$. Similarly, the assembly sequences of the alignment skill, handover skill, insert1 skill, and insert2 skill generated respectively are $4 \rightarrow 4 \rightarrow 8 \rightarrow 9 \rightarrow 8 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 9 \rightarrow 10 \rightarrow 4 \rightarrow 4$ $1 \rightarrow 3 \rightarrow 7 \rightarrow 5 \rightarrow 1 \rightarrow 1 \rightarrow 3 \rightarrow 2 \rightarrow 3 \rightarrow 3$ $1 \rightarrow 5 \rightarrow 1 \rightarrow 1 \rightarrow 1$ $15 \rightarrow 5 \rightarrow 7 \rightarrow 3 \rightarrow 3 \rightarrow 3$.

B. EVALUATION

To prove the effectiveness of the proposed algorithm, we use our data and compare the proposed algorithm with different methods, including the Gaussian mixture regression (GMR) [16] and the local weight regression (LWR) [35]

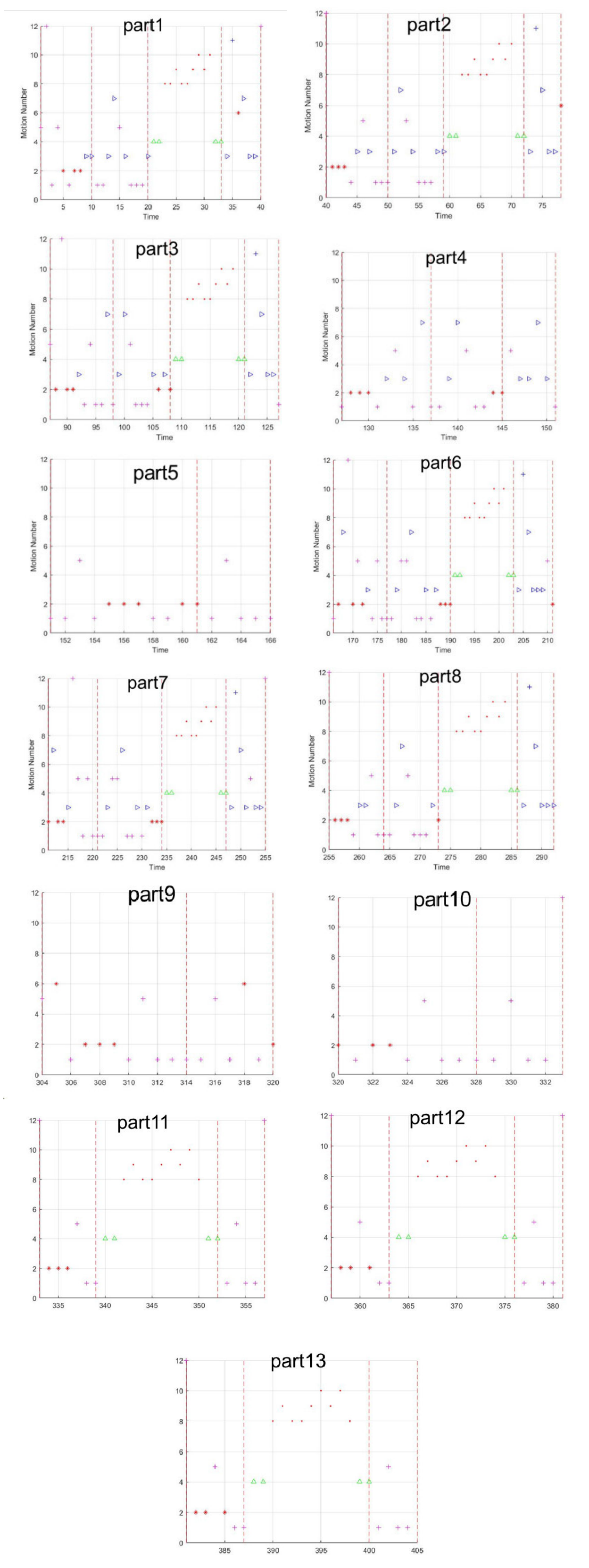


FIGURE 7. Motion process data of 13 kinds of parts.

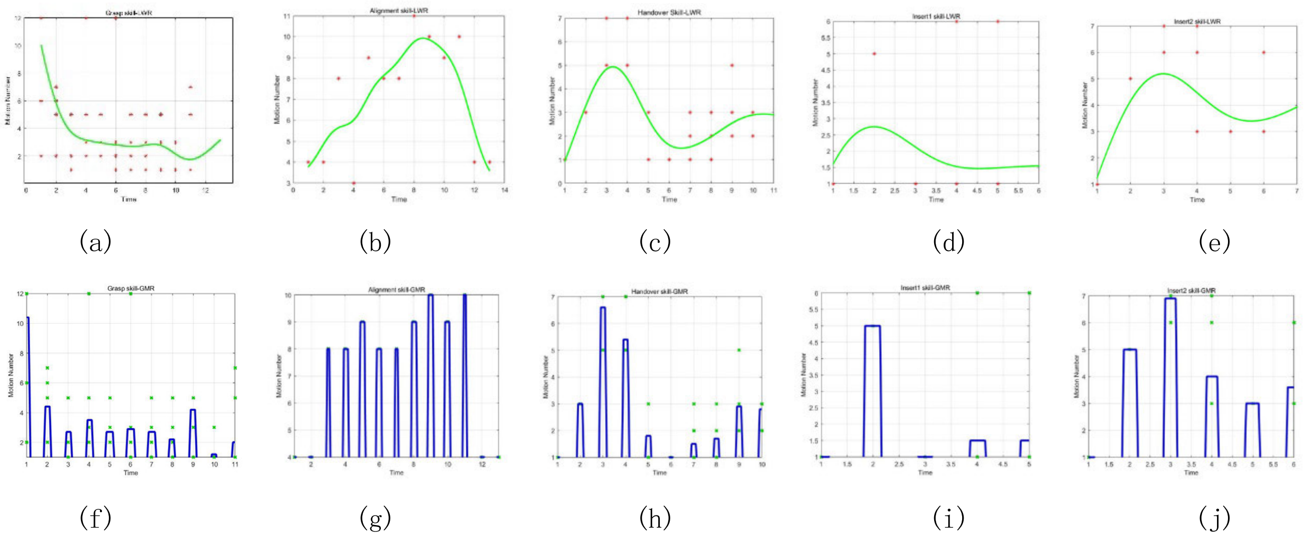


FIGURE 8. (a)–(e) show the learning results of five skills based on the LWR method, respectively. The values (green lines) are computed by LWR model based the demonstration data (red dots). (f)-(j) are the learned results of five skills based on GMR.

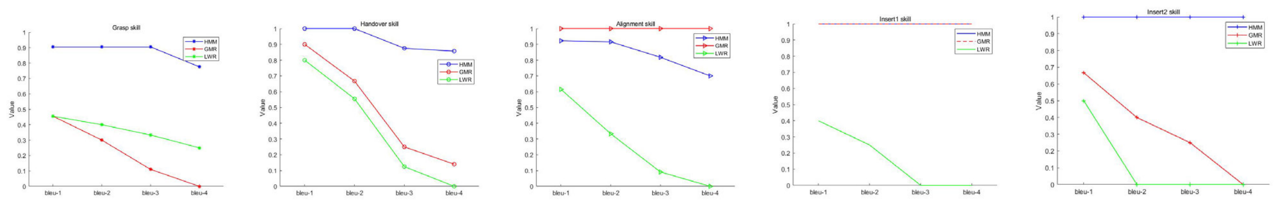


FIGURE 9. The precision score of five skills.

algorithm. We used the same data for learning in Section 5.1. Based on the above two methods, the action sequences of five different skills are generated. During the GMR experiment, the number of Gaussian mixture models for the five skills were set to 35, 13, 20, 7 and 10, respectively. The calculation results of GMR and LWR are shown in Figure 8. Finally, the action sequences of the five skills obtained based on the LWR method are 2-3-3-3-2-2-2-3-3-2-2, 1-3-5-5-3-1-2-2-3-3,4-5-6-6-7-8-9-10-10-9-8-6-4, 2-3-2-1-1, 11-4-5-4-4-3. Similarly, based on the GMR method, the action sequences of the five skills are 10-4-3-4-3-3-3-2-4-1-2, 1-3-7-5-2-1-1-2-3-3,4-4-8-8-9-8-8-9-10-9-10-4-4,1-5-1-1-1, 11 -5-7-4-3-4.

Based on the above results, we will evaluate and compare the generated data. BLEU (bilingual evaluation understudy) [36] is a popular evaluation method, used to evaluate the difference between two strings, and can also be used to evaluate the accuracy of the generated assembly sequences. The BLEU evaluation standard here is to compare the degree of similarity between the generated assembly action sequence and the sample action sequence. BLEU compares and counts the number of n-gram that co-occur in the generated action sequence and the sample action sequence. His calculation formula is as follows

$$BLEU = BP \times \exp(\sum_{n=1}^{N} \omega_n \log p_n) \tag{11}$$

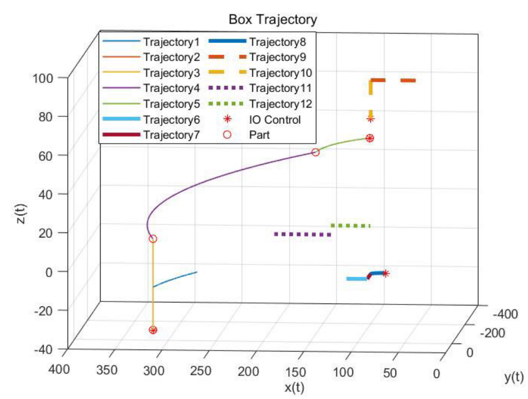


FIGURE 10. Trajectories of each module.

where BP is the brevity penalty, ω_n is positive weight, p_n represents a modified precision score of *n-gram*. When the similarity between the generated sequence and these sample sequences is higher, the *BLEU* is higher. Through calculation, the *BLEU* calculation results of five assembly skills are obtained as shown in Fig. 9.

Obviously, the method can accurately characterize the knowledge from the data. The precision score of the proposed method is closed to 0.9. On the contrary, the precision score of GMR and LWR are much lower than 0.9. The hidden Markov

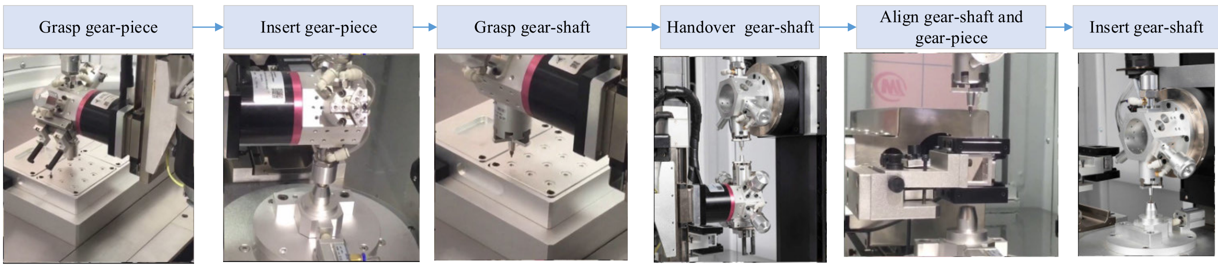


FIGURE 11. The assembly process of gear-piece and gear-shaft.

model can better represent the relationship between the front and back actions, while GMR and LWR are more sensitive to different actions at the same time. It is the reason why our method outperforms two comparative methods.

C. ACCURACY POSITION AND TRAJECTORY

Additionally, a small amount of human knowledge in the process of generating and deploying to the actual assembly system is integrated in this paper. In the process of skill execution, the precise position is combined with the action primitive in the form of parameters. The precise position of the parts in the tray is calculated by image processing. The grasping position, collision avoidance points and release position of MJRM are obtained by teaching through the teach pendant. The trajectory control of each module is realized in an end-to-end strategy. In addition, some processes are carried out manually in the preparation stage. Such as, the position where the CAM extends, the posture adjusted by the 3 DOFs MAM, the movement position of the AMM. The sample trajectory of each module is shown in the Fig. 10.

The trajectories from 1 to 5 are the movement trajectories of the end of the robot, which are described as fellow: 1) the robot moves directly above the part based on vision guidance; 2) the robot moves down and grabs the part based on the calibration; 3) the robot grabs the part and moves it up to the specified position; 4) the robot carries the parts and moves to the space avoidance point; 5) the robot carries the parts and moves to the position to be assembled.

The trajectories from 6 to 8 represent the movement trajectories of the micro-motion platform, which respectively represent movement in the X-axis direction, movement in the Y-axis direction, and rotation around the Z-axis.

The trajectories 9 and 10 respectively represent the movement of the assembly execution manipulator in the X and Z directions.

The trajectories 11 and 12 respectively represent the movement of the coaxial alignment module. The trajectories 11 represents the horizontal movement of the entire module, and the trajectories 12 represents the horizontal movement of the prism relative to the camera.

The asterisk indicates the IO action, and the circle indicates that the track carries the parts to be assembled.

The manipulators and fixtures of the modular assembly system are simply replaced to assemble new parts. For the assembly of gear and pinion parts, the instruction stream combined with the generated action sequence and position parameters is deployed to the assembly system after fine-tuning. The part of the gear-piece and gear-shaft is shown in Fig. 6(c). The automated assembly of the gear-piece and gear-shaft is effectively implemented, and the fit clearance of the gear shaft is 10 micrometers. The assembly process of this part is shown in Fig. 11, which verifies the feasibility of the method.

VII. CONCLUSION

In this paper, we develop a reconfigurable assembly system to realize the modularization of hardware and software, ensuring rapid response to new types of parts. A new skill learning framework is presented, which regards the skill as a serial of assembly motion primitive for modular assembly system. Our primitives include not only common motion commands such as axis movement, IO control, and robot movement, but also force feedback control and visual feedback control. Additionally, in the aspect of high-level structured demonstration and learning, the proposed method is successfully applied in a modular assembly system for new task. Though the demonstration of the assembly process of 13 parts, a data set was created to train the proposed algorithm in this paper. The results show that the proposed multiple HMM method can effectively express motion constraints. The comparative experiment shows that the proposed method has high accuracy for the representation of the motion sequence of the modular assembly system. Finally, the assembly experiment of the gear and shaft was verified through our platform, which showed that the algorithm has good application value and can reach an assembly accuracy of 10 microns. In summary, the assembly skills learning framework proposed in this article can provide a good example for assembly tasks in factories.

In the future work, we will focus on precision automated assembly for diverse parts. Through the collection of a large amount of data in actual and simulation, the quality of algorithm generation is further improved. In addition, some sim-to-real learning techniques, such as deep reinforcement learning, are used to solve the problem of precise position calibration of each module, thereby improving the ability of the assembly system to learn action strategies.

REFERENCES

[1] Y. Wang, Y. Jiao, R. Xiong, H. Yu, J. Zhang, and Y. Liu, "MASD: A multimodal assembly skill decoding system for robot programming by demonstration," *IEEE Trans. Autom. Sci. Eng. (from July 2004)*, vol. 15, no. 4, pp. 1722–1734, Oct. 2018.

[2] F. Li, Q. Jiang, W. Quan, S. Cai, R. Song, and Y. Li, "Manipulation skill acquisition for robotic assembly based on multi-modal information description," *IEEE Access*, vol. 8, pp. 6282–6294, 2020.

[3] J. Luo, E. Solowjow, C. Wen, and J. A. Ojea, "Reinforcement learning on variable impedance controller for high-precision robotic assembly," *Int. Conf. Robot. Autom. (ICRA)*, Montreal, QC, Canada, 2019, pp. 3080–3087.

[4] H. Herrero, A. A. Moughlbay, J. L. Outón, and D. Sallé, "Skill based robot programming: Assembly, vision and workspace monitoring skill interaction," *Neurocomputing*, vol. 255, pp. 61–70, Sep. 2017.

[5] F. Qin, D. Xu, D. Zhang, and Y. Li, "Robotic skill learning for precision assembly with microscopic vision and force feedback," *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 3, pp. 1117–1128, Jun. 2019.

[6] S. Calinon, "Learning from demonstration (programming by demonstration)," in *Encyclopedia of Robotics*, M. Ang, O. Khatib, and B. Siciliano, Eds. Berlin, Germany: Springer, 2018.

[7] G. E. Hovland, P. Sikka, and B. J. McCarragher, "Skill acquisition from human demonstration using a hidden Markov model," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Minneapolis, MN, USA, Apr. 1996, pp. 2706–2711.

[8] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook of Robotics*. Berlin, Germany: Springer, 2008, pp. 1371–1394.

[9] M. Kyrarini, M. A. Haseeb, D. Ristić-Durrant, and A. Gräser, "Robot learning of industrial assembly task via human demonstrations," *Auto. Robots*, vol. 43, no. 1, pp. 239–257, Jan. 2019.

[10] H.-D. Zhang, S.-B. Liu, Q.-J. Lei, Y. He, Y. Yang, and Y. Bai, "Robot programming by demonstration: A novel system for robot trajectory programming based on robot operating system," *Adv. Manuf.*, vol. 8, no. 2, pp. 216–229, Jun. 2020.

[11] Y. Gu, W. Sheng, C. Crick, and Y. Ou, "Automated assembly skill acquisition and implementation through human demonstration," *Robot. Auto. Syst.*, vol. 99, pp. 1–16, Jan. 2018.

[12] C. Zeng, C. Yang, J. Zhong, and J. Zhang, "Encoding multiple sensor data for robotic learning skills from multimodal demonstration," *IEEE Access*, vol. 7, pp. 145604–145613, 2019.

[13] G. Gorjup, G. P. Kontoudis, A. Dwivedi, G. Gao, S. Matsunaga, T. Mariyama, B. MacDonald, and M. Liarokapis, "Combining programming by demonstration with path optimization and local replanning to facilitate the execution of assembly tasks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 1885–1892.

[14] J. Duan, Y. Ou, J. Hu, Z. Wang, S. Jin, and C. Xu, "Fast and stable learning of dynamical systems based on extreme learning machine," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 6, pp. 1175–1185, Jun. 2019.

[15] N. Liu, X. Zhou, Z. Liu, H. Wang, and L. Cui, "Learning peg-in-hole assembly using Cartesian DMPs with feedback mechanism," *Assembly Automat.*, vol. 40, no. 6, pp. 895–904, 2020.

[16] S. Calinon and A. Billard, "Statistical learning by imitation of competing constraints in joint space and task space," *Adv. Robot.*, vol. 23, no. 15, pp. 2059–2076, Jan. 2009.

[17] W. He, C. Xue, X. Yu, Z. Li, and C. Yang, "Admittance-based controller design for physical human–robot interaction in the constrained task space," *IEEE Trans. Automat. Sci. Eng.*, vol. 17, no. 4, pp. 1937–1949, Oct. 2020.

[18] Y. Li, F. Qin, S. Du, D. Xu, and J. Zhang, "Vision-based imitation learning of needle reaching skill for robotic precision manipulation," *J. Intell. Robotic Syst.*, vol. 101, no. 1. p. 22, 2021.

[19] R. Hanifi Elhachemi Amar, L. Benchikh, H. Dermeche, O. Bachir, and Z. Ahmed-Foitih, "Trajectory reconstruction for robot programming by demonstration," *Int. J. Electr. Comput. Eng. (IJECE)*, vol. 10, no. 3, p. 3066, Jun. 2020.

[20] G. Xiang and J. Su, "Task-oriented deep reinforcement learning for robotic skill acquisition and control," *IEEE Trans. Cybern.*, vol. 51, no. 2, pp. 1056–1069, Feb. 2021.

[21] Y. Mollard, T. Munzer, A. Baisero, M. Toussaint, and M. Lopes, "Robot programming from demonstration, feedback and transfer," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Hamburg, Sep./Oct. 2015, pp. 1825–1831.

[22] M. Diab, M. Pomarlan, D. Beßler, and A. Akbari, "SkillMaN–A skill-based robotic manipulation framework based on perception and reasoning," *Robot. Auton. Syst.*, vol. 134, Dec. 2020, Art. no. 103653.

[23] K. Lee, Y. Su, T.-K. Kim, and Y. Demiris, "A syntactic approach to robot imitation learning using probabilistic activity grammars," *Robot. Auto. Syst.*, vol. 61, no. 12, pp. 1323–1334, Dec. 2013.

[24] K. Bernardin, K. Ogawara, K. Ikeuchi, and R. Dillmann, "A sensor fusion approach for recognizing continuous human grasping sequences using hidden Markov models," *IEEE Trans. Robot.*, vol. 21, no. 1, pp. 47–57, Feb. 2005.

[25] A. Kramberger, R. Piltaver, B. Nemec, M. Gams, and A. Ude, "Learning of assembly constraints by demonstration and active exploration," *Ind. Robot: Int. J.*, vol. 43, no. 5, pp. 524–534, Aug. 2016.

[26] C. S. Chen, S. K. Chen, C. T. Lin, and C. Lai, "Sequential motion primitives recognition of robotic arm task via human demonstration using hierarchical BiLSTM classifier," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 502–509, Apr. 2021.

[27] K. Sasabuchi, N. Wake, and K. Ikeuchi, "Task-oriented motion mapping on robots of various configuration using body role division," 2020, *arXiv:2007.08750*. [Online]. Available: http://arxiv.org/abs/2007.08750

[28] C. Deng, X. Ji, C. Rainey, J. Zhang, and W. Lu, "Integrating machine learning with human knowledge," *iScience*, vol. 23, no. 11, Nov. 2020, Art. no. 101656.

[29] L. Roveda, M. Magni, M. Cantoni, D. Piga, and G. Bucca, "Human–robot collaboration in sensorless assembly task learning enhanced by uncertainties adaptation via Bayesian optimization," *Robot. Auton. Syst.*, vol. 136, Feb. 2021, Art. no. 103711.

[30] S. Niekum, S. Chitta, A. Barto, B. Marthi, and S. Osentoski, "Incremental semantically grounded learning from demonstration," in *Robotics: Science and Systems*, vol. 9. Berlin, Germany, Jun. 2013, doi: 10.15607/rss.2013.ix.048.

[31] K. Ramirez-Amaro, E. Dean-Leon, F. Bergner, and G. Cheng, "A semantic-based method for teaching industrial robots new tasks," *KI-Künstliche Intelligenz*, vol. 33, no. 2, pp. 117–122, Jun. 2019.

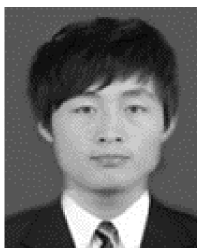
[32] C. Shao, X. Ye, J. Qian, Z. Zhang, and D. Zhu, "Robotic precision assembly system for microstructures," *Proc. Inst. Mech. Eng., I, J. Syst. Control Eng.*, vol. 234, no. 8, pp. 948–958, Sep. 2020.

[33] R. Lioutikov, G. Maeda, F. F. Veiga, K. Kersting, and J. Peters, "Learning attribute grammars for movement primitive sequencing," *Int. J. Robot. Res.*, vol. 39, no. 1, pp. 21–38, Nov. 2019.

[34] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[35] L. Peternel, T. Petric, and J. Babic, "Human-in-the-loop approach for teaching robot assembly tasks using impedance control interface," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Seattle, WA, USA, May 2015, pp. 1497–1502.

[36] P. Kishore, S. Roukos, T. Ward, and W. J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proc. ACL*, 2002, pp. 311–318.



PENGYUE GUO received the M.S. degree in mechanical engineering from the Shenyang Institute of Technology, Shenyang, China, in 2017. He is currently pursuing the Ph.D. degree in aerospace manufacturing with the Beijing Institute of Technology, Beijing, China. His research interests include vision servo, intelligent programming, learning from demonstration, and skill learning.

ZHIJING ZHANG received the B.S. and M.S. degrees from the Beijing Institute of Technology, China, in 1975 and 1982, respectively.

He is currently a Professor with the Beijing Institute of Technology. He is also the Chief Scientist of the "973" Project of Precision Assembly and the Director of the Chinese Precision Assembly Committee of the Chinese Institute of Production–Chinese Mechanical Engineering Society (CIP-CMES). His research interests include precision microminiature complex machining technology, precision micro-structure flexible automation, intelligent assembly technology, and precision assembly theory.

YAN LIU received the B.S. degree in industrial design from Shandong Jianzhu University, Jinan, China, in 2013, and the M.S. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2020. His research interests include intelligent programming and intelligent assembly.

YUJUN LIU received the B.S. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2019, where he is currently pursuing the M.S. degree with the School of Machinery and Vehicle. His research interests include precision assembly and intelligent robot.

DONGSHENG ZHU received the M.S. degree in mechanical engineering from the North China University of Science and Technology, Tangshan, China, in 2018. He is currently pursuing the Ph.D. degree in mechanical engineering with the Beijing Institute of Technology, Beijing, China. His research interests include intelligent assembly, precision assembly method, and assembly system design.

CHAO SHAO received the M.S. and Ph.D. degrees in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2015 and 2019, respectively. He is currently a Senior Engineer with the Beijing Institute of Control Engineering, Beijing. His research interests include mechanical structure design, space robot, and intelligent assembly.