Hindawi
Journal of Robotics
Volume 2020, Article ID 3196983, 11 pages
https://doi.org/10.1155/2020/3196983

Research Article

A New Type of Industrial Robot Trajectory Generation Component Based on Motion Modularity Technology

Zhaoming Liu,[1,2,3] **Nailong Liu**,[1,2,3] **Hongwei Wang**,[1,2] **Shen Tian**,[1,2] **Ning Bai**,[1,2] **Feng Zhang**[1,2] **and Long Cui**[1,2]

[1]*State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China*
[2]*Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China*
[3]*University of Chinese Academy of Sciences, Beijing 100049, China*

Correspondence should be addressed to Zhaoming Liu; liuzhaoming@sia.cn

Received 30 April 2019; Revised 2 August 2019; Accepted 13 August 2019; Published 28 January 2020

Academic Editor: Shahram Payandeh

Motion modularity is the main method of motion control for higher animals. That means the complex movements of the muscles are made up of basic motion primitives, and the brain or central nervous system does not care about the specific details of the movement. However, the industrial robot control system does not adopt the technical roadmap of motion modularity, it generates complex trajectories by providing a large number of sampling points. This approach is equivalent to using the brain to directly guide the specific movement of the muscle and has to rely on a faster Fieldbus system to obtain complex motion trajectories. This work constructs a modularized industrial robot trajectory generation component based on Dynamic Movement Primitives (DMP) theory. With this component, the robot controller can generate complex trajectories without increasing the sampling points and can obtain good trajectory accuracy. Finally, the rationality of this system is proved by simulations and experiments.

1. Introduction

Motion modularity is an idea of decomposing complex motion into simple motion modules. That means complex motions are composed of some motion primitives. Studies in motor neuroscience have shown that the nervous systems of higher animals are controlled by a modular approach. The central nervous system achieves coordinated control of multiple muscles and multiple joints by combining modules that represent basic motion [1]. Neuroscience research on motion modularity has achieved many results in muscle synergy [2], motion primitive understanding [3], motion pattern learning [4], and motion modeling [5].

Since higher animals rely on modular methods to complete motion control, motion modularity technology should also be the future of robot control. However, currently the robot control system does not adopt the motion modularity technology roadmap, it adopts another centralized control technology roadmap which enhances the capability of Fieldbus to improve control effect. We will introduce this in detail in Section 3. Actually, scholars in the field of robotics have conducted long-term research on motion modularity. Professor Roger Brockett of Harvard University first proposed this idea in robotics. He pointed out that the motion of robots has the characteristics of the hybrid system, and can drive a robot to generate continuous motion by symbol string and real-valued functions. He proposed Motion Description Language (MDL) method [6, 7]. After that, Motion Description Language was combined with the dynamic system to establish a control method for the inverted pendulum choreography robot [8]. MDL has been applied in teleoperation system [9]. Since then, MDL has also been applied to the trajectory planning and control of mobile robots [10]. Based on the function decomposition theory of Hilbert space, this work decomposes the trajectory of the robot to obtain the motion primitives. The modular trajectory is sent to the mobile robot and good results are achieved. This method has been promoted and applied to the control of the industrial robot via mapping the robot joint trajectory to the motion primitives by function space decomposition [11]. Then, a small number of motion primitives are sent to the robot joint and the joint reproduces the trajectory by combining the primitives. Through this technology, the purpose of not reducing the accuracy of the trajectory but reducing the real-time communication frequency of the robot is realized. However, this method has problems such as insufficient generalization ability of the motion primitives and the unstable number of primitive elements, and it cannot be deployed in the actual industrial robot control system.

In this paper, the Dynamic Motion Primitive (DMP) technology is used to expand the MDL method, and the shortcomings of the previous work are solved. In the past, the DMP method was mainly used for imitative learning of robots which was on the task level. To reduce the data without compromising the accuracy of trajectory, the novelty of this work is that we apply the DMP method on the level of joint trajectory generation firstly based on the motion modularity idea and a new industrial robot trajectory generation component is added to the industrial robot controller.

The following chapters will first sort out the Dynamic Motion Primitive theory, and then elaborate on the industrial robot trajectory generation component based on the motion modularity technology. Finally, the corresponding simulation and experimental results are introduced.

2. Basis of Dynamic Movement Primitives

The Dynamic Movement Primitives (DMP) method was first proposed by Professor Stefan Schaal of the University of Southern California and is a method for robot trajectory control and planning. This method improves the generalization ability of the robot motion primitives and can regress the trajectory better. It has many applications in the research of robot imitative learning [12], human-robot collaboration [13], and human motion modeling [14]. Besides, this method has been applied to neurosurgical science research by scientists, shows how movement primitives can be used to learn appropriate muscle excitation patterns and to generalize effectively to new reaching skills [15]. This achievement combines the study of motion modularity in neuroscience and robotics.

The basic idea of the DMP theory is to use a relatively simple and stable system to parameterize the expression of complex trajectories and then adjusts the system through a nonlinear function, so that the trajectory of the system exhibits the desired characteristics. It inherits the advantages of linear systems: convergence to the attracting domain, robustness to interference, and time independence. The addition of a force function allows the system to generate arbitrary motion trajectories. There are two main types of DMP models: discrete and rhythmic [16]. Discrete systems are based on attraction points and rhythms are based on limit cycles. The trajectory of the robot manipulator is essentially a mapping from time to robot configuration. In the actual system, it is a discrete motion through numerous intermediate points. Therefore, the discrete DMP is selected as the basis of manipulator trajectory control.

The DMP system consists of two parts: a second-order system and a force function system. The second-order system produces a basic motion that converges to the target location. In the control of electromechanical systems, most of the models

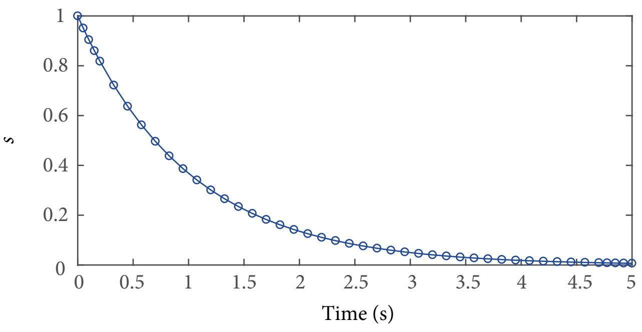

Figure 1: The first-order system $\dot{s} = -\alpha s$. The original state of s is 1 and converges to 0.

can be described by the spring-damping model. Therefore, we use the spring-damping model as the basic second-order system in the DMP. The force function is a supplement to the second-order system and is used to fit the motion pattern of the original trajectory by producing an arbitrarily smooth motion trajectory. Using the spring-damped system to construct DMP, we can get the basic equations of the DMP system:

$$\tau\begin{bmatrix}\dot{v}\\ \dot{x}\end{bmatrix} = \begin{bmatrix}\alpha(\beta(g-x)-v)+f(t)\\ v\end{bmatrix}. \tag{1}$$

In the basic equation, the force function $f(t)$ is essentially a function approximator that uses some regression method to approximate the original trajectory. The DMP system is made to converge to a given target while producing a trajectory that satisfies the requirements and is sufficiently smooth. However, it does not ensure that the system converges to the target point g after the force function is added to the basic equation. Therefore, a gating term should be added that makes the force function converge to 0 while the system reaches the target g. The first-order system $\dot{s} = -\alpha s$ has such a property (Figure 1), so it is considered to use the variable s as a gating term. In addition, the force function $f(t)$ is a function that depends on the system time; it is not a time-independent autonomous system. We should make the force function an autonomous system that does not depend on the system time t. Therefore, the independent variable t of the force function can be replaced by the s of the gating term. The entire force term can be expressed as $sf(s)$. Thus, a complete DMP system can be expressed in the form of the following equation:

$$\tau\begin{bmatrix}\dot{v}\\ \dot{x}\\ \dot{s}\end{bmatrix} = \begin{bmatrix}\alpha(\beta(g-x)-v)+sf(s)cc\\ v\\ -\alpha s\end{bmatrix} init\cdot\begin{bmatrix}0\\ x_0\\ 1\end{bmatrix} attr\cdot\begin{bmatrix}0\\ g\\ 0\end{bmatrix}. \tag{2}$$

Among Equation (2), α and β are the spring rate and damping coefficient respectively related to the spring system, g indicates the target position of the system, x, v , $\dot{v}$ are the system position, velocity and acceleration states respectively. The parameter τ is a scaling factor that controls the speed of motion. $sf(s)$ is the force function with gating term and represented by a phase parameter.

The forced function $f(s)$ is a function approximator that uses the regression method to approximate the original trajectory. There are many algorithms of function regression that can be summarized into two unified models, which are mixture of linear models (3) and weighted sum of basis functions (4) [17].

$$f(X) = \sum_{n=1}^{N} \phi(X, \theta_n) \cdot \left(a_n^T X + b_n\right), \tag{3}$$

$$f(X) = \sum_{n=1}^{N} \phi(X, \theta_n) \cdot \omega_n. \tag{4}$$

In this paper, we use the local weighted regression (LWR) algorithm that belongs to the mixture of linear models to approximate. According to the characteristics of the LWR algorithm and the DMP system, the unified model can be simplified, and the specific expression form of the function approximator is obtained:

$$f(s) = \frac{\sum_{(i=1)}^{N} \psi_i(s)\omega_i}{\sum_{(i=1)}^{N} \psi_i(s)}, \tag{5}$$

where ψ is a standard Gaussian function:

$$\psi_i(s) = \exp\left(-\frac{\left(s - c_i\right)^2}{2\sigma_i^2}\right). \tag{6}$$

c_i is the center of the Gaussian function, and σ_i is the width of the Gaussian function, which is a known parameter given in advance by the user. The entire DMP system only has the weight parameter ω_i which needs to be solved by prior data.

To solve the optimal parameter ω of the system, we need to pre-plan the end-effector trajectory of the manipulator according to the task requirements, and then calculate the reference trajectory of the robot joint according to the kinematics and dynamics model of the robot. After that, the n sets state point data $y_d, \dot{y}_d, \ddot{y}_d$ are sampled on the trajectory with a fixed sampling period. Finally, the sampling point data are substituted into the original system equation, and the force function can be obtained.

$$f_d = \ddot{y}_d - \alpha(\beta(g - y_d) - \dot{y}_d). \tag{7}$$

According to the above Equation (7) and the force function $f(s)$, the minimum error criterion $J = \sum \left(f_d(s) - f(s)\right)^2$ is selected to find the optimal weight of the system ω_i

$$\omega_i = \frac{s^T \psi_i f_d}{s^T \psi_i s}, \tag{8}$$

among it

$$s = \begin{bmatrix} x_{t_0} \\ \vdots \\ x_{t_n} \end{bmatrix}, \qquad \psi_i = \begin{bmatrix} \psi_i(t_0) & \cdots & 0 \\ 0 & \ddots & 0 \\ 0 & \cdots & \psi_i(t_n) \end{bmatrix}. \tag{9}$$

At this point, the theoretical preparation of the DMP controller is completed. Figure 2 shows the whole process of DMP algorithm from the training of motion primitives to the implementation.

3. Trajectory Generation Component Based on Motion Modularity Technology

This section will take an actual robot control system as an example to analyze the traditional industrial robot control system in detail. Figure 3 shows the SIA-SmartPainting, a painting system for vehicle maintenance developed by our team. Its controller is a typical centralized industrial robot control system. The system visually models the car's door, generates a paint path in the upper controller and performs interpolation operation, then sends intermediate points of trajectory to the joint through EtherCAT bus. According to the characteristics of the robot control task, the robot control system can be divided into four levels from top to bottom: planning layer, trajectory layer, communication layer, and joint layer. At the planning layer, the SIA-SmartPainting system visualizes the painted door to create a motion path of the robot's end-effector. The trajectory layer performs operations such as inverse kinematics, trajectory interpolation, and time parameterization. It maps the path in Cartesian space to the joint space and adds time parameters. It also plans reasonable velocity and acceleration through interpolation and optimization algorithms. After obtaining the reference trajectory of the joint space, the relevant data are transmitted to the communication layer, and the intermediate point of the reference trajectory is sent to the joint layer by the Fieldbus system of the communication layer. Since the communication layer is a real-time component, the closed-loop control of the manipulator is also carried out in this layer. To ensure the smoothness of the trajectory and the stability of the spray gun while painting operation, the communication layer needs to perform closed-loop control of the position and speed of the joint. To improve the communication rate, we adopt the high-speed EtherCAT bus, and the real-time communication frequency can reach 1 kHz, that is, the communication layer performs closed-loop control with a period of 1 ms. The bottom layer of this control system is the joint layer. This layer receives the reference signal sent from the Fieldbus, controls the joint motor motion according to the reference signal, and feeds the motor status to the upper controller in real-time.

Through the detailed analysis of a typical industrial robot control system, we can find that the control system of an industrial robot is mainly composed of two kinds of hardware: the upper controller and the joint motor driver. The upper controller is responsible for the control of the whole robot arm and the joint motor driver is responsible for the motion control of the joint motor. The two parts rely on the Fieldbus system to communicate. In the traditional industrial robot control system, the path of the robot is generated in the planning layer; the trajectory layer is responsible for the inverse kinematics and interpolation calculation; the communication layer sends the reference points of the trajectory to the joint in real-time; the joint layer is the motor control unit, which controls the specific motion of the joint motor according to the reference signal. This control system is essentially a discrete control which discretizes the joint's trajectory and then sends the discrete intermediate points of trajectory to the joint controller. The characteristic of this system is that all the motion details

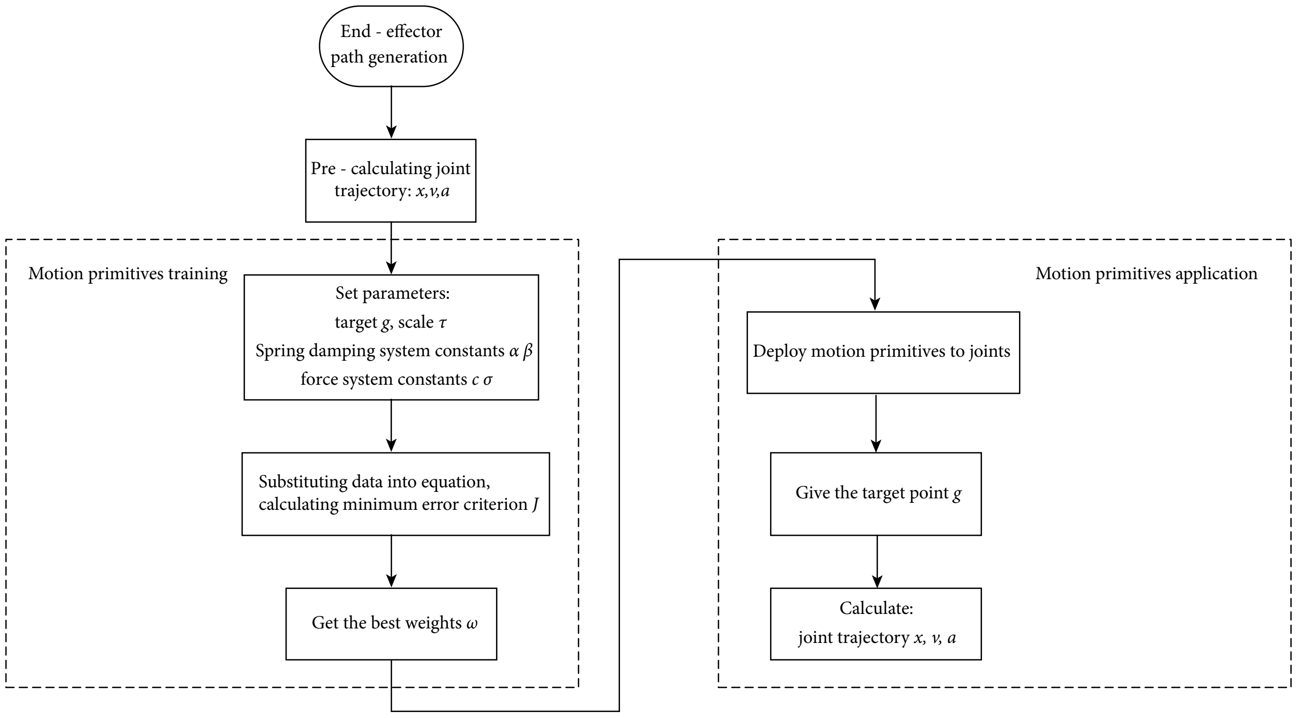
End - effector path generation
Pre - calculating joint trajectory: x,v,a
Motion primitives training
Set parameters:
target g, scale τ
Spring damping system constants α β
force system constants c σ
Substituting data into equation, calculating minimum error criterion J
Get the best weights ω
Motion primitives application
Deploy motion primitives to joints
Give the target point g
Calculate:
joint trajectory x, v, a

FIGURE 2 : The DMP algorithm flowchart.

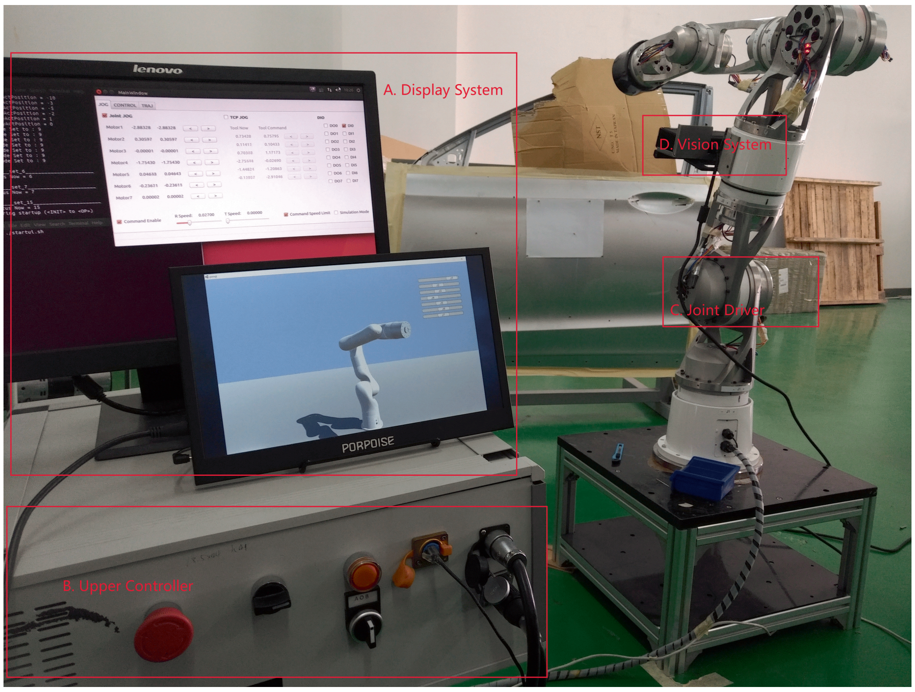

FIGURE 3: The SIA-SmartPainting system: (a) the display system, (b) the upper controller, (c) the joint driver, (d) the vision system.

of the joint are generated by the upper controller, and the joint itself is only an execution unit without any autonomy. This kind of control system is a centralized structure. Under this structure, if we want to improve the trajectory accuracy of the robot, we need to increase the number of sampling points of the interpolation point, and the communication rate of the Fieldbus determines the accuracy of the trajectory. In connection with the idea of motion modularity, we hope to establish a modularized trajectory generation component. This component, with motion primitives in joints, can improve the trajectory accuracy and the performance of the robot without increasing the sampling points and communication rate.

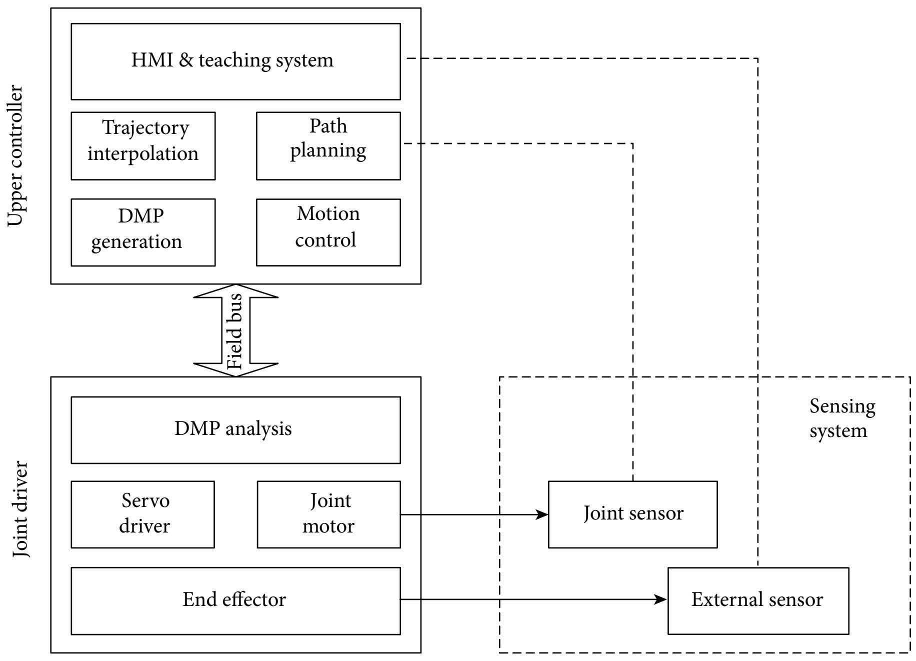

Figure 4: The structure of controller system with modularized component.

a) Function of layers

Layers \ Architecture	Traditional architectrue	Modularized architecture
Planning layer	Path planning	Path planning
Trajectory layer	Trajectory generation	Trajectory generation motion primitives generation
Communication layer	Trajectory control	Motion primitives deploying trajectory control
Joint layer	Joint motor control	Motion primitives analysis joint motor control

b) Fieldbus

Fieldbus \ Architecture	Traditional architectrue	Modularized architecture
Type of fieldbus	EtherCAT	CAN or RS232
Communication frequency	High(typical 1kHz)	Low (typical 100Hz)

c) Control mode

Control \ Architecture	Traditional architectrue	Modularized architecture
Feedback	Position, Velocity	Position
Feedforward		Position, Velocity
Control cycle	Short (typical 1 ms)	Long (typical 10 ms)

Figure 5: Centralized structure vs modularized structure.

There has been a lot of research on modular robots, but the current modularity mainly refers to modular joints, which are modular in mechanical and electrical aspects. That is to say, only hardware is modular and there is no software modularity currently. We hope to use motion modularity technology in the structure of the system, such that the motion is generated by the motion primitives. Therefore, we have made a trajectory generation component based on the DMP theory that generates motion in a modular way, improves joint autonomy, and achieves good motion performance without transmitting a large number of intermediate points of the trajectory. As can be seen from the structure diagram (Figure 4), the modularized system has one more component for DMP model generation and analysis than the traditional control system. This new system first precalculates the robot trajectory and then trains the basic DMP motion primitives with the obtained trajectories in advance. After obtaining the optimal parameters, we deploy these primitives to the joint driver. When the robot is running, the joint itself calculates the feedforward information of the trajectory via motion primitives. The upper controller only needs to control the position of the joint trajectory at a very low frequency, so that an accurate and smooth trajectory can be obtained. This system implements the control that is feedforward and supplemented by feedback.

Figure 5 shows the similarities and differences after adding the new trajectory generation component to the robot control system. The hierarchy is similar, but there are differences in

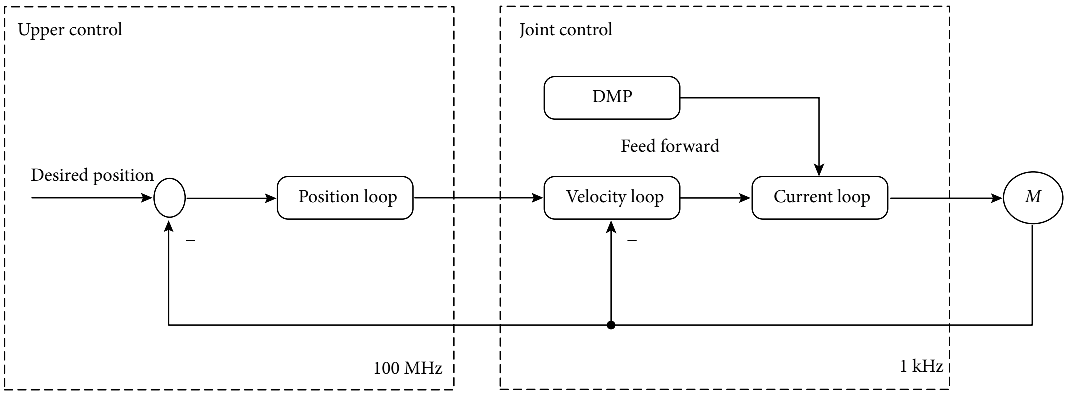

Figure 6: The block diagram of controll system.

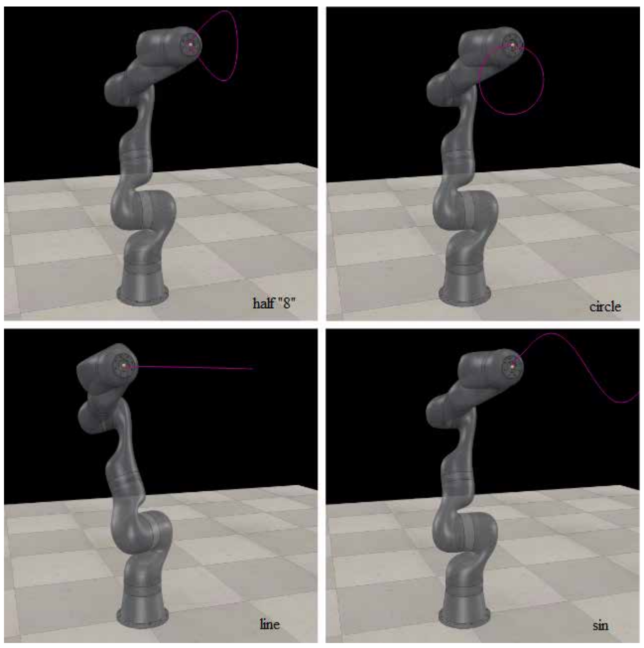

Figure 7: SIA-SmartPainting robot in V-REP. The robot is controlled to reproduce the pink end-effector trajectory.

the specific functions of each layer. The planning layer is no different from the traditional controller and is still responsible for the path planning. The trajectory layer is responsible for the trajectory solving tasks and the generation of motion primitives. Specifically, this layer should calculate the reference trajectory of the joint space according to the optimization index of the system and the kinematics and dynamics model of the robot. Then generates a DMP model based on the previously obtained trajectory to complete the mapping from trajectory to motion primitives. The communication layer is responsible for deploying the motion primitives to the robot joint driver in advance. Thus, during the motion of the manipulator, the joint driver can generate a feedforward signal of the trajectory according to the predeployed motion primitives.

Traditionally, to obtain a highly accurate and smooth trajectory, the upper controller generally performs reference trajectory sampling and control with a period of 1 ms. The advantage of this new method is that the reference trajectory is generated autonomously by the joint, and the upper controller can greatly improve the control period. For example, the joint's position loop can be controlled with a period of 10 ms, but the trajectory accuracy and smoothness can also be realized. Figure 6 shows the relationship between feedforward and feedback of this kind of control strategy. With the modularized component, the robot control system can complete complex trajectory with a small number of sampling points, achieve the control effect of high-speed Fieldbus with the low-speed Fieldbus, and improve the accuracy of the trajectory without increasing the real-time data.

4. Simulations

To verify the feasibility of the trajectory generation method in the robot control system, we need to simulate the system at first. The simulation environment is built by V-REP and MATLAB platforms. V-REP is a robot integrated simulation environment with the robot and environment model. We can quickly develop and research robot algorithms through embedded scripts, plug-ins or remote APIs [18]. The V-REP contains the dynamics engine that supports the Unified Robot Description Format (URDF) file. Importing a URDF file with the dynamic characteristics and inertia information of the robot, the robot can be dynamically simulated and the joints can be controlled. The developer can also directly adjust the joint PID parameters. We will simulate the robot body and joint driver through V-REP. The user can operate robot joints and obtain feedback by calling V-REP's Remote APIs in MATLAB. Therefore, we write a program to simulate the robot's controller in MATLAB.

This simulation uses the SIA-SmartPainting robot model as the analysis object, and controls the robot to reproduce the four types of end-effector trajectory in different ways (Figure 7), then compare and analyze the relevant characteristics. The specific versions of the simulation software are V-REP 3.5 and MATLAB 2014a. The simulation adopts the synchronous mode of V-REP, the simulation period is 5 ms. Programs are written in MATLAB which are applied to both traditional and modularized system architectures:

Table 1: The weights of force function.

Half "8" (1e+4)					Circle (1e+4)				
−2.2979	−2.3695	−2.4435	−2.3718	−2.0582	0.1681	0.2688	0.443	0.607	0.7142
−1.4371	−0.486	0.7424	2.1014	3.3706	0.7254	0.6066	0.3462	−0.0209	−0.3859
4.3119	4.7234	4.4585	3.4363	1.7055	−0.5654	−0.3867	0.1416	0.7143	0.845
−0.4195	−2.3042	−3.3686	−3.8401	−4.2306	0.2923	−0.6525	−1.3983	−1.4788	−1.1544
Line (1e+3)					**Sinusoid**				
−0.5787	−0.5692	−0.5505	−0.5259	−0.4957	−161.373	−167.742	−172.901	−168.012	−147.138
−0.4586	−0.4132	−0.3576	−0.2897	−0.2073	−105.795	−41.5869	44.489	147.56	260.2086
−0.1076	0.0125	0.1564	0.3278	0.5308	374.8938	483.7187	572.6412	615.5687	579.3059
0.77	1.0503	1.3771	1.7377	1.996	441.335	212.2539	−43.9249	−201.26	−204.401

Table 2: The error analysis of simulation.

Type of trajectory	System archiceture	RMSE
Half "8"	Traditional	0.0154
Half "8"	Modularized	0.0084
Circle	Traditional	0.0112
Circle	Modularized	0.0091
Line	Traditional	0.0070
Line	Modularized	0.0068
Sinusoid	Traditional	0.0115
Sinusoid	Modularized	0.0100

(1) Traditional architecture: This scheme is applied to the control system of the traditional architecture. The upper system controls the position loop of the joint with a control period of 50 ms and there is no modularized tajectory generation component on joint.

(2) Modularized architecture: This scheme is applied to the control system of the modularized architecture. The upper system controls the position loop of the joint with a control period of 50 ms and there is modularized tajectory generation component on joint. The joint is simulated at the period of 5 ms to generate feedforward signal for the position and velocity loops.

In order to make a quantitative analysis of the reproducibility of the trajectory, we define the pose error of the end-effector trajectory. Since the pose matrix of the robot belongs to the Special Euclidean Group $SE(3)$, which is not closed to addition, it is considered to map the end pose matrix to the corresponding Lie Algebra $se(3)$ for error analysis and comparison. Let the desired pose matrix be T_d, and the actual matrix of trajectory is T_g, so that the actual trajectory points are in one-to-one correspondence with the desired trajectory points. Then the error of the i-th point on the trajectory is defined as:

$$e_i = \left\| \ln\left(T_d\right)^{\vee} - \ln\left(T_g\right)^{\vee} \right\|_2. \quad (10)$$

That is the two norms of the difference between the two Lie Algebras. Thus, we can define the Root Mean Square Error (RMSE) of the two trajectories:

$$RMSE\,(d, g) = \sqrt{\frac{1}{n}\sum_{i=n}^{n} e_i^2}. \quad (11)$$

Table 1 shows the weights of force function in different tajectory type. Table 2 shows the analysis of the simulation results and Figure 8 shows the position curve of the robot in Cartesian coordinates. In the four sets of simulations, the robot used the same joint PID parameters. It can be seen that adopting the modularized architecture, better control effects can be obtained with the same control period and the same amount of data. Simulation analysis proves the feasibility of the trajectory generation component based on DMP technology.

5. Experiments

5.1. Experimental Platform. In order to verify the practical ability of the motion modularity technology in the industrial robot control system, we built a principle verification experimental platform consisting of an upper controller and a joint motor driver (Figure 9). The hardware of the upper controller is an computer with a real-time Linux system. The software system is mainly based on ROS and has been developed packages such as trajectory generation, DMP calculation, and communication. The trajectory generation package is developed based on the pilz industrial robot trajectory generation library [19]. The DMP calculation package is developed based on the DMPBBO calculation library [20]. Figure 10 shows the overall node architecture and information flow of the experimental platform, the blue dashed box indicates the corresponding tool used by the node.

The joint driver is based on the STM32 platform and is a brushless motor driver with DMP analysis. In order to improve the processing speed and the stability of the motor control, the driver is divided into two independent parts of low voltage side and high voltage side. Each part is assigned an independent CPU. The high voltage side uses the STM32F303 chip, a chip with an internal integrated amp unit which is very practical and suitable for motor control applications. The chip is responsible for sampling armature current and PWM generation. The low voltage side is responsible for control and operation, which is handled by an STM32F405 chip. The F4 chip contains a floating-point arithmetic unit, which can provide good support for mathematical operations. Therefore, the brushless motor FOC control, PID calculation, and DMP analysis are performed on the low voltage side. Figure 11 is a

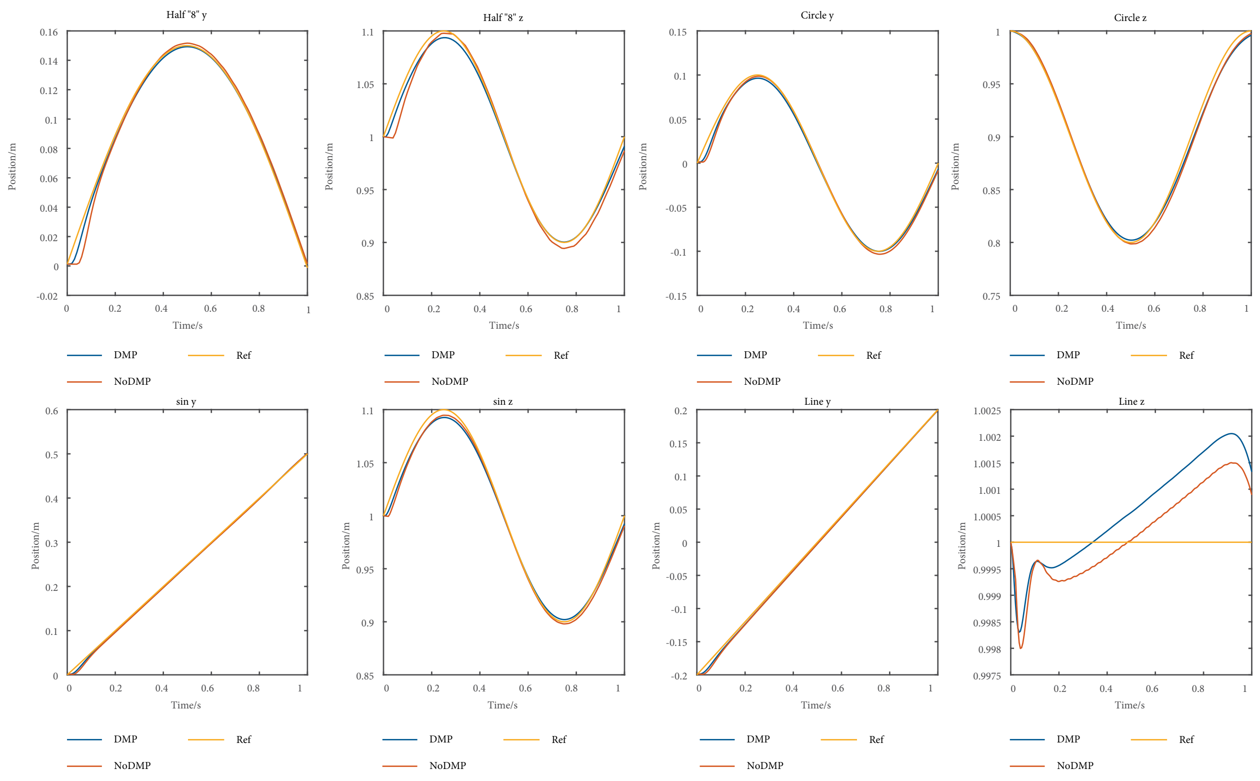

Figure 8: Position curve of the four simulations in Cartesian space.

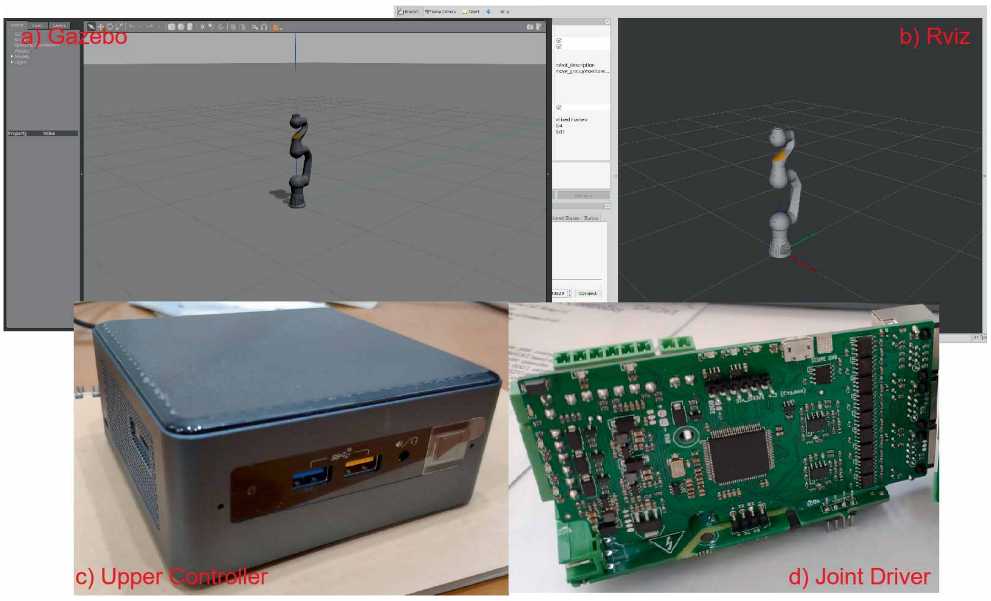

Figure 9: Experimental platform. (a) The Gazebo for simulation; (b) Rviz the UI sytem; (c) upper controller with Linux; (d) joint driver based on STM32.

functional diagram of the driver unit. Figure 12 is the joint motor driver based on STM32.

5.2. Experimental Results. At this time, a single joint motion experiment was performed. Firstly, the robot model is controlled to run the previous four types of end-effector trajectory in the simulation environment of the upper control system, and then the trajectory data of each joint are collected. We use the joint trajectory data collected in the simulation environment to train the DMP model. After that, we send the parameters to the joint driver in advance. The joint motor driver stores the motion primitives after receiving the DMP's parameters. As soon as the target point is sent to the motor driver, the motor begins to run. During the motion, the upper

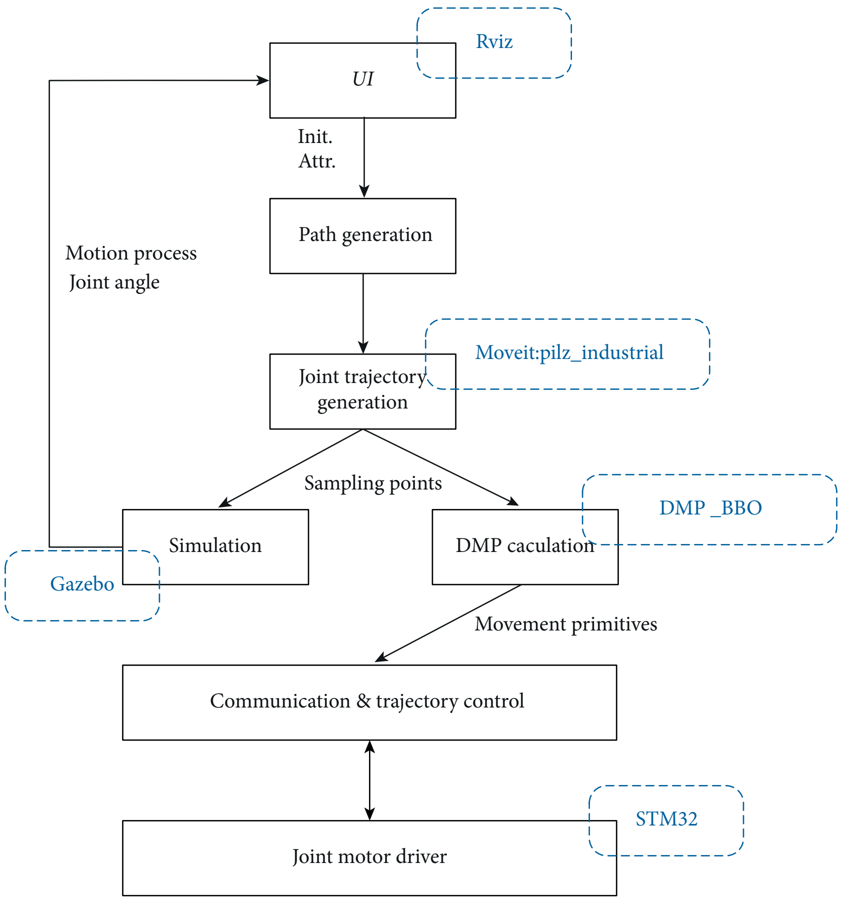

Figure 10: Experimental platform framework. Blue dashed boxes are the main software packages on the system.

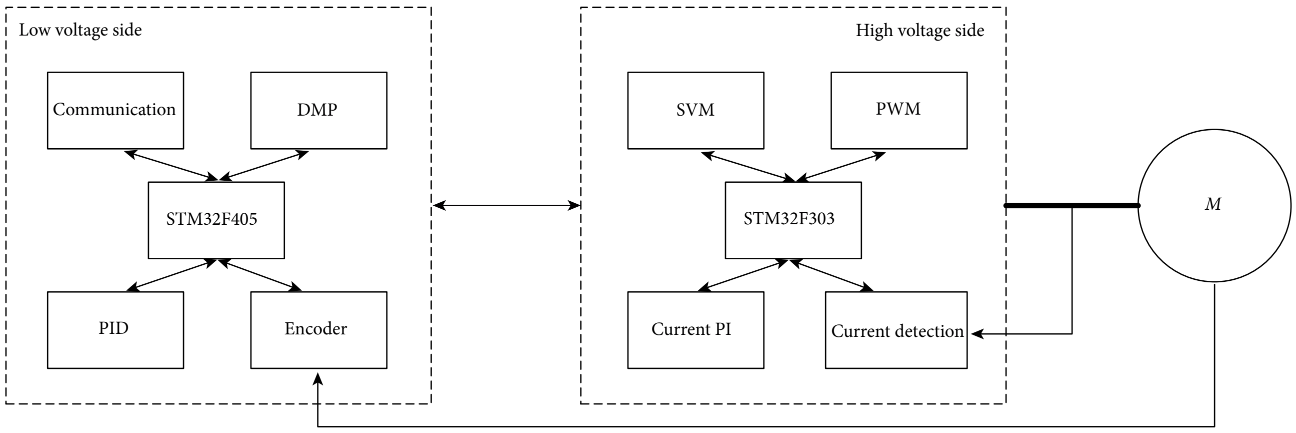

Figure 11: Function module of joint motor driver.

controller controls the position loop of the motor in a period of 50 ms, which is longer than the joint feedforward given period (feedforward period is 5 ms). In the middle of the two control signals, the CPU on the low voltage side of the driver calculates the motion feedforward values (position, velocity, and acceleration) in real time. These feedforward signals are inputs of the corresponding control loop of the driver.

Since we cannot show the results of every joint, we only show the typical trajectory curves of joint 2 (Figure 13), it can be seen that the system achieves a good reconstruction of the trajectory. This experiment proves the effectiveness of the modularized control system for industrial robots.

6. Discussion

The current industrial robot control systems are centralized, and the upper control system is responsible for all the details

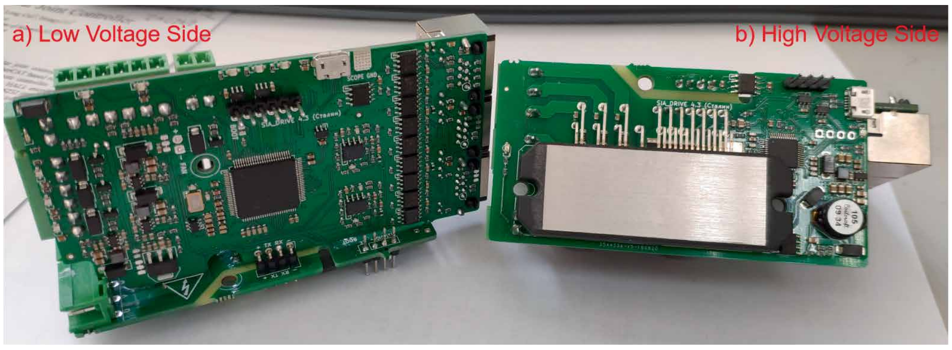

Figure 12: Joint motor driver. (a) The low voltage side; (b) the high voltage side.

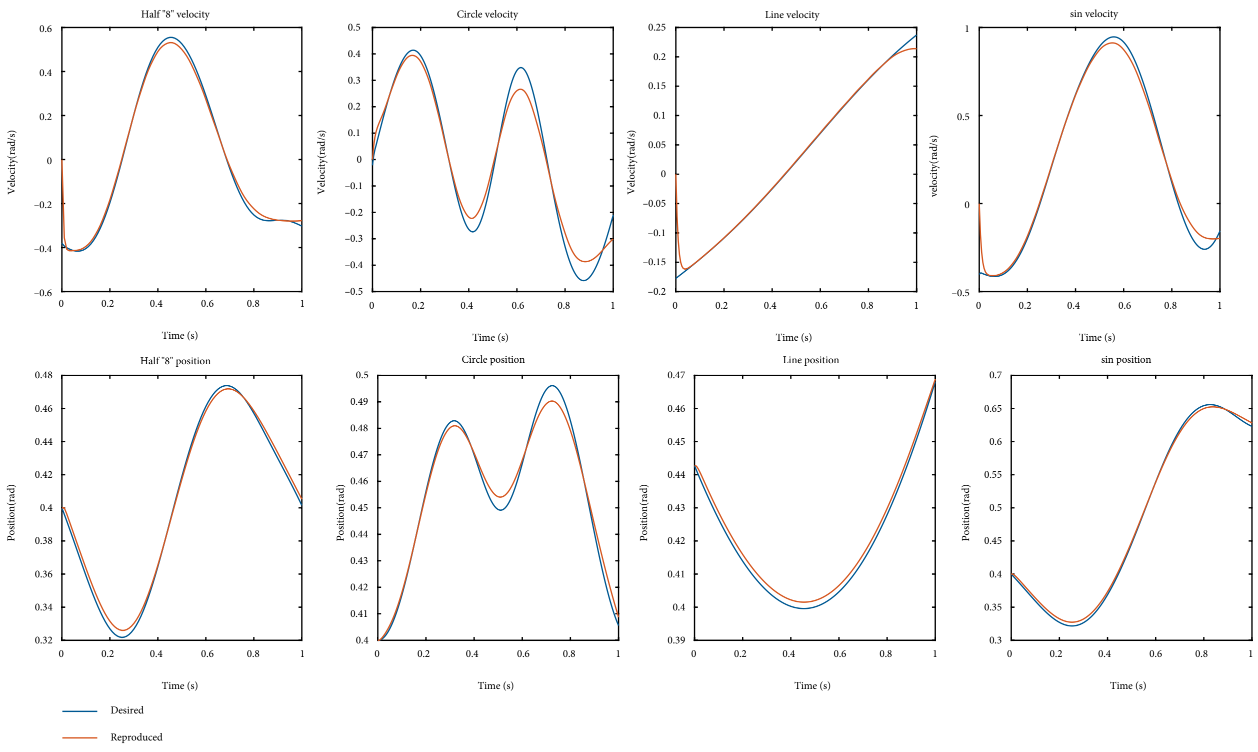

Figure 13: The experimental data of joint 2.

of the motion, just as humans use the brain to guide the motion of each muscle. However, neuroscience research has shown that higher animals generate and control motion in a modular manner. Therefore, we hope that the robot control system can also have the same ability of motion modularity, which combining complex motion by simple primitives and improving joint's autonomy. Based on the DMP theory, this work builds a modularized trajectory generation component for the industrial robot controller. Firstly, the method of motion primitive generation is constructed by DMP theory. Then the function and benefits of the components are analyzed. Finally, the feasibility of this system is proved by simulation and experiment.

The significance of the motion modularity component in practical work can be listed as following:

(1) The autonomy of the joint has been improved. The joint automatically generates the feedforward information of the trajectory through the motion primitives and does not need to be provided all the motion details by the upper controller. The trajectory accuracy which can be achieved with 1 ms as the control cycle currently can be achieved via 10 ms through this component. That means reducing the real-time requirements of the main controller. We do not need to use the expensive real-time operation system.

(2) A low-speed Fieldbus system can be used. Since the communication cycle of the upper controller is greatly improved, the high-speed Fieldbus system is no longer necessary. For example, the robot can also achieve good motion performance after replacing the EtherCAT bus with the CAN bus.

(3) The robot can implement one master multi slaves control mode. Because of reducing the control frequency of the upper controller. The upper controller

can greatly reduce the frequency of the feedback control. That means the ability of multirobot cooperation will be improved. For example, on the production line, one upper controller can be responsible for multiple manipulators and the operations are synchronized by the same upper controller. That will make it easier for multi-robot to work together.

This study also has the following shortcomings, which need to be improved in the future:

(1) The clock synchronization of the joints has not been considered. Clock synchronization is a key issue in multiaxis control. This work only uses the local clock of the joint itself, and does not have multiaxis clock synchronization. Clock synchronization will be the focus of subsequent research.

(2) The characteristics of motion primitives are not delved deeper. Currently the system obtains motion primitives by precalculating the trajectory, without analyzing the characteristics of the motion primitives. We hope to extract the characteristics of the motion primitives, classify them and establish a one-one mapping between primitives and trajectories.

Data Availability

The data and all other info used or produced in this project will be available for the publisher.

Conflicts of Interest

The authors declare that there are no conficts of interest regarding the publication of this paper.

Acknowledgments

This work is funded by International Thermonuclear Experimental Reactor (ITER) Project under Grant 2012GB102005.

References

[1] A. d'Avella, "Modularity for motor control and motor learning," *Progress in Motor Control*, Springer, pp. 3–19, 2016.

[2] Naveen Kuppuswamy and Christopher M. Harris, "Do muscle synergies reduce the dimensionality of behavior?," *Frontiers in Computational Neuroscience*, vol. 8, p. 63, 2014.

[3] I. Zelman, M. Titon, Y. Yekutieli, S. Hanassy, B. Hochner, and T. Flash, "Kinematic decomposition and classification of octopus arm movements," *Frontiers in Computational Neuroscience*, vol. 7, p. 60, 2013.

[4] S. A. Overduin, A. d'Avella, J. M. Carmena, and E. Bizzi, "Muscle synergies evoked by microstimulation are preferentially encoded during behavior," *Frontiers in Computational Neuroscience*, vol. 8, p. 20, 2014.

[5] M. Sartori, L. Gizzi, D. G. Lloyd, and D. Farina, "A musculoskeletal model of human locomotion driven by a low dimensional set of impulsive excitation primitives," *Frontiers in Computational Neuroscience*, vol. 7, p. 79, 2013.

[6] W. B. Roger, "Hybrid models for motion control systems," *Essays on Control*, Springer, pp. 29–53, 1993.

[7] B. Roger, "Language driven hybrid systems," in *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, IEEE, pp. 4210–4214, 1994.

[8] H. Li and R. W. Brockett, "DMDL for choreography," in *Proceedings of the 44th IEEE Conference on Decision and Control*, IEEE, pp. 4016–4021, Spain, 2005.

[9] J. Hua, H. Li, Y. Wang, and X. Ning, "MDL-based control method for mobile robot with randomly varying time-delay," in *IEEE International Conference on Robotics and Automation*, IEEE, pp. 1772–1777, China, 2011.

[10] Eugene Frank Gargas III, "Generation and use of a discrete robotic controls alphabet for high-level tasks," Georgia Institute of Technology, 2012.

[11] L. Zhaoming, L. Nailong, W. Qing, and C. Long, "Motion description language for trajectory generation of a robot manipulator," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, IEEE, pp. 1950–1955, China, 2017.

[12] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," in *Advances in Neural Information Processing Systems*, pp. 1547–1554, Canada, 2003.

[13] W. Liu, D. Chen, L. Zhang, "Trajectory generation and adjustment method for robot manipulators in human-robot collaboration," *Jiqiren Robot*, vol. 38, no. 4, pp. 504–512, 2016.

[14] J. F.-S. Lin, M. Karg, and D. Kulić, "Movement primitive segmentation for human motion modeling: a framework for analysis," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 3, pp. 325–339, 2016.

[15] E. Rückert and A. d'Avella, "Learned parametrized dynamic movement primitives with shared synergies for controlling robotic and musculoskeletal systems," *Frontiers in Computational Neuroscience*, vol. 7, p. 138, 2013.

[16] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 2013.

[17] F. Stulp and O. Sigaud, "Many regression algorithms, one unified model: a review," *Neural Networks*, vol. 69, pp. 60–79, 2015.

[18] E. Rohmer, S. P. N. Singh, and M. Freese, "V-rep: a versatile and scalable robot simulation framework," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, pp. 1321–1326, Japan, 2013.

[19] Pilz, "pilz industrial motion – industrial trajectory generation for moveit!," 2018.

[20] F. Stulp, "DmpBbo – a c++ library for black-box optimization of dynamical movement," *Primitives*, 2014.